UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Jure Flander

ZASNOVA IN IZVEDBA PROTOTIPA ZA OMREŽNI INTELIGENTNI HLADILNIK

DIPLOMSKO DELO NA UNIVERZITETNEM ŠTUDIJU

Mentor: doc. dr. Mira Trebar
Somentor: doc. dr. Mojca Ciglarič

Ljubljana, 2008

IZJAVA O AVTORSTVU

diplomskega dela

Spodaj podpisani Jure Flander,

z vpisno številko 63010024,

sem avtor diplomskega dela z naslovom:

ZASNOVA IN IZVEDBA PROTOTIPA ZA OMREŽNI INTELIGENTNI HLADILNIK

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Mire Trebar in somentorstvom doc. dr. Mojce Ciglarič.
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela.
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki „Dela FRI“.

V Ljubljani, dne 11.12.2008 Podpis avtorja:

Zahvala

Zahvaljujem se mentorici, doc. dr. Miri Trebar, in somentorici, doc. dr. Mojci Ciglarič, za vodenje pri pisanju diplomske naloge. Zahvaljujem se tudi vsem ostalim, ki so kakorkoli pripomogli k nastanku tega dela.

Kazalo

Seznam uporabljenih kratic

AGC	Automatic Gain Control
AM	Amplitude Modulation
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
ASP	Active Server Pages
CSS	Cascading Style Sheet
EEPROM	Electrically Erasable Programmable Read-Only Memory
GUI	Graphical User Interface
HF	High Frequency
HVAC	Heating, Ventilation and Air Conditioning
IRQ	Interrupt Request
ISO	International Organization for Standardization
JRE	Java Runtime Environment
JSP	JavaServer Pages
LCD	Liquid Crystal Display
LF	Low Frequency
PHP	PHP Hypertext Preprocessor
PM	Phase Modulation
RAD	Rapid Application Development
RF	Radio Frequency
RFID	Radio-Frequency Identification
RSSI	Received Signal Strength Indication
SQL	Structured Query Language
SRD	Short Range Devices
SWT	Standard Widget Toolkit
UHF	Ultra Hight Frequency
UID	Unique Identifier
URL	Uniform Resource Locator
USB	Universal Serial Bus
XHTML	Extensible Hypertext Markup Language

Povzetek

V diplomskem delu sta predstavljeni zasnova in izvedba prototipa za omrežni inteligentni hladilnik. Ta sistem na podlagi tehnologije radiofrekvenčne identifikacije (RFID) avtomatsko vodi evidenco artiklov v hladilniku. Omogoča nam prikaz te vsebine, obvestil in predlogov obrokov glede na zalogo. Upravljamo ga preko zaslona, občutljivega na dotik. Poleg tega je povezan v internet, preko katerega je omogočen oddaljen dostop do posameznih funkcionalnosti.

Strojni del sistema predstavlja osebni računalnik z zaslonom na dotik, na katerega so priključeni še digitalna tehtnica in dva čitalca RFID. Programski del pa sestavljajo servis, aplikacija z grafičnim vmesnikom ter spletna aplikacija. Za implementacijo teh komponent so bile uporabljene tehnologije Java in .NET.

Predstavljena je tudi kratka primerjava s klasičnim hladilnikom in nekaj idej za nadaljnje delo.

Ključne besede

Radiofrekvenčna identifikacija, inteligentni hladilnik, servis, grafični uporabniški vmesnik, spletna aplikacija.

Abstract

In this thesis, a prototype design and implementation of a networked intelligent refrigerator is presented. Based on RFID technology, a system automatically maintains refrigerator's content database. It features displaying contents, alerts and meal suggestions based on database records of tracked items. A touch-screen LCD panel is used for main user interaction. Besides that, the refrigerator is connected to the internet allowing remote access to the some of the functionalities.

The hardware equipment of the system is composed of a computer using touch screen display, two RFID readers and a digital weighing scale. Software consists of a system service, main GUI application and a web application. Implementation was done using Java and .NET technologies.

There is also presented a short comparison to a classic refrigerator, and some ideas for future work.

Key words

Radio frequency identification, smart refrigerator, service, graphical user interface, web application.

1 Uvod

Hladilnik je naprava, ki je prisotna praktično v vsakem gospodinjstvu. V taki obliki, kot jo poznamo danes, obstaja že več kot 80 let. V tem času se je hladilnik razvijal samo v smeri učinkovitejšega hlajenja, zmanjšanja porabe energije in uporabe modernih tehnologij. V osnovi pa ves čas ostaja le neke vrste omara za hrambo živil [11, 18].

Z napredkom v tehnologiji se je izoblikoval koncept pametne hiše, ki nam v prvi vrsti ponuja večje udobje. Pametna hiša naj bi vsebovala tehnologije za samodejno upravljanje ter nadzor različnih naprav in sistemov. Izkorišča sodobne komunikacijske in informacijske tehnologije za povezavo posameznih elementov in njihovo integracijo v celovit sistem. Omogoča širokopasovno komunikacijo z zunanjem svetom ter navsezadnje energetsko učinkovito in varno bivanje [7].

Hladilnik kot element pametne hiše naj ne bi bil več le kos pohištva. Beležil naj bi svojo vsebino, na podlagi tega pa nam ponudil nove funkcionalnosti, kot so: obveščanje o preteku roka uporabe živil, predlaganje jedilnika glede na zalogo in potrebe (dieta), pomoč pri nakupovanju in podobo.

V današnjem času so vsi artikli na trgu označeni s črtnimi kodami. V prihodnosti pa se pričakuje, da bodo le-te zamenjale nalepke RFID. Glavna prednost nalepk RFID je v tem, da za identifikacijo ni potreba vidna ali fizična povezava s čitalcem. Z uporabo tehnologije RFID bi bilo mogoče hladilnik nadgraditi tako, da bi sam avtomatsko vodil evidenco vsebine ter tako deloval v skladu s konceptom pametne hiše.

Cilj diplomske naloge je izdelati prototip inteligentnega hladilnika, ki temelji na tehnologiji RFID. Hladilnik naj omogoča avtomatsko vodenje vsebine, prikaz vsebine, obvestil in predlogov obrokov glede na vsebino. Poleg tega naj bo ta hladilnik povezan v internet, preko katerega naj se omogoči oddaljen dostop do posameznih funkcionalnosti.

2 Pametna hiša

Bivalni prostor je za človeka ključnega pomena. V preteklosti nas je dom le ščitil pred vplivi žive in nežive narave. Z razvojem tehnologije pa je bivalno okolje v zgradbah postajalo vse bolj neodvisno od zunanjosti. V domove so si utrle poti številne naprave, brez katerih si življenja danes ne znamo več predstavljati: ogrevalni sistem, telefon, hladilnik, televizija itd. Njihova skupna značilnost je preprosto in zanesljivo delovanje brez možnosti medsebojne komunikacije in zadovoljive stopnje samozavedanja. Ker tako zgrajeno okolje ne more skrbeti samo zase, je za njegovo delovanje in vzdrževanje potrebno precej truda.

Pametna hiša, kot jo vidimo na sliki 1, naj bi z integracijo samodejnega upravljanja in nadzora nad različnimi področji avtomatizacije klasično domovanje razširila v enoten sistem s centralnim upravljanjem preko interneta in lokalnega omrežja.

Slika 1: Pametna hiša.

Med funkcije takega doma spadajo: kontrola vstopa, regulacija temperatur, upravljanja alarmov, odpiranja oken, zaves, pomoči pri kuhanju, shranjevanju hrane v hladilniku, branja elektronske pošte in zahtevnejše, kot so: sledenje osebam, analiza počutja stanovalcev – domači zdravnik, izdelava družinskega albuma, pomoč in svetovanje pri pomembnih odločitvah. Možnosti je ogromno. Pomembno pa je, da sistem z vsemi funkcijami v hiši služi človeku, ga razbremenjuje, mu nudi razvedrilo in udobje.

2.1 Inteligentni hladilnik

Zadnjih 10 let se različni proizvajalci trudijo uspeti na trgu s slabimi približki inteligentnega hladilnika. Več ali manj so bili ti poizkusi in so še neuspešni. Ti produkti so praktično le klasični hladilniki z v vrata vgrajenim tabličnim računalnikom (tablet PC). Prav zaradi teh manipulacij proizvajalcev je bila pred kratkim narejena raziskava, v kateri so večjo skupino ljudi povprašali, kaj naj bi po njihovem mnenju inteligentni hladilnik bil oziroma katere funkcionalnosti naj bi omogočal. Večina se jih strinja z definicijo v naslednjem odstavku, prav tako so te osebe navedle, da bi tak hladilnik tudi kupile, če bi že obstajal [12, 16].

Inteligentni hladilnik kot del koncepta pametne hiše je hladilnik, ki se zaveda svoje vsebine. Torej sam avtomatsko vodi evidenco vsebine. Na podlagi tega pa ponuja funkcionalnosti, kot so: obveščanje o preteku roka uporabe živil, podajanje predlogov jedilnika glede na zalogo, pomoč pri nakupovanju izdelkov, vodenje diete, možnost oddaljenih pregledov vsebine in podobno.

Čeprav tak inteligentni hladilnik komercialno še ni na voljo, pa poteka na tem področju več raziskav. Izdelani so tudi že prvi delujoči prototipi, na osnovi katerih se je že razvil koncept inteligentnega hladilnika. Ti prototipi za avtomatsko vodenje vsebine hladilnika izkoriščajo tehnologijo radiofrekvenčne identifikacije (RFID), ki je predstavljena v nadaljevanju. Njihov namen je predvsem spremljanje zalog, prikaz podatkov o shranjenih artiklih in sporočanje nakupovalne liste [8, 9].

2.2 Radiofrekvenčna identifikacija RFID

Tehnologija radiofrekvenčne identifikacije (RFID), ki je ime dobila po radijskem območju frekvenčnega spektra, ki ga uporablja za prenos informacije, je v svetu znana že dobrih petdeset let. Pojavila se je že v obdobju druge svetovne vojne, v šestdesetih pa so v komercialne namene že razvili sistem, ki je obvaroval izdelke v trgovinah pred krajo in se uporablja še danes. Z leti se je tehnologija izboljšala, zaradi preprostosti pa se je kmalu

uveljavila tudi na drugih področjih in jo danes nezavedno uporabljamo za različne namene, kot so kontrola pristopa, plačevanje cestnine in podobno. Tehnologija ima velike možnosti za uporabo v trgovinah, saj je poceni in jo bo mogoče v kratkem uporabiti za vsak izdelek. Zamenjala bo črtne kode, hkrati pa bo povečala sledljivost in preglednost nad zalogami v oskrbovalnih verigah [1, 19].

V osnovi je sistem RFID sestavljen iz treh komponent. Prva komponenta je značka RFID, ki je povezana z objektom, ki ga želimo identificirati. Naslednja je čitalec RFID, s katerim se podatki iz značke berejo. Zadnja pa je aplikacija, ki prebrane podatke pretvori v uporabno informacijo. Shemo takega sistema prikazuje slika 2.

Slika 2: Shema sistema radiofrekvenčne identifikacije RFID.

2.2.1 Tipi sistemov RFID

Sistemi RFID se v osnovi delijo glede na a) način napajanja značke [6, 17]:

- Aktivni RFID

  Aktivna značka RFID poleg antene in čipa vsebuje tudi baterijo, ki napaja vezje oddajnika. To značkam bistveno poveča ceno in dimenzije, nudi pa določene prednosti, kot so večja moč oddajanja, daljši domet ter zanesljivejše delovanje v neprijaznem okolju (bližina kovine in tekočine).

- Pasivni RFID

  Pasivne značke za razliko od aktivnih nimajo svojega napajanja, temveč potrebno energijo za delovanje dobijo neposredno od signala, ki se inducira v anteni. Sprejet izmenični signal se usmeri ter dovede do čipa, ki se vzbudi in začne delovati. Pasivne značke imajo zaradi odsotnosti baterije precej nižjo ceno in dimenzije od aktivnih, vendar posledično veliko krajši domet in manjšo odpornost na napake.

ter b) glede na dolžino dometa:

- Sistemi z dometom do 1 m

  Sisteme z dometom do enega metra označujemo kot sklopljene sisteme. Skoraj vsi sklopljeni sistemi temeljijo na induktivnem (magnetnem) sklopu med čitalcem in oddajnikom, zato jim pravimo tudi induktivni sistemi. Obstajajo tudi kapacitivni sistemi, ki pa jih na tržišču skoraj ne najdemo več. Delovanje je teoretično mogoče na katerikoli frekvenci od 0 do približno 30 MHz. Prenos energije med nosilcem podatkov in čitalcem je močan, zato lahko uporabimo cenejše mikroprocesorje z neoptimalno porabo energije. Sklopljeni sistemi se uporabljajo v aplikacijah, ki ne zahtevajo velikega dometa ali pa so predmet poostrenega nadzora.

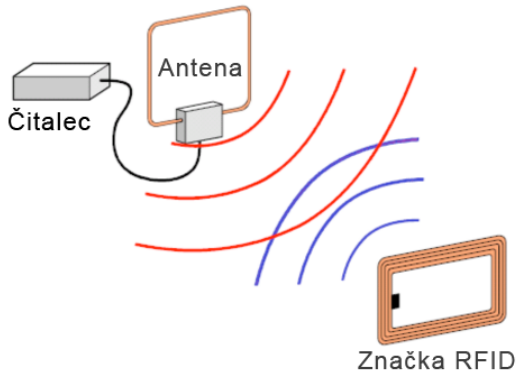
- Sistemi z dometom preko 1 m

  Za sisteme z dometom preko enega metra se uporabljajo ultra visoke frekvence (UHF) okoli 900 MHz ter v nekaterih aplikacijah tudi frekvence iz mikrovalovnega področja nekaj GHz. Vsi taki sistemi za prenos informacije uporabljajo elektromagnetne valove. S pasivnimi oddajniki je trenutno mogoče doseči domete do 10 m, medtem ko se aktivni oddajniki uporabljajo za domete več 10 m.

V tem delu nas zanima le sistem RFID s pasivnimi značkami dometa do 1 m, zato se v nadaljevanju predpostavlja, da govorimo o tej inačici.

2.2.2 Kako deluje pasivni sistem RFID

Način delovanja radiofrekvenčne identifikacije je zelo preprost. Najbolje ga lahko opišemo kot sistem, ki ga sestavljata značka (predstavlja jo integrirano vezje s pripadajočo anteno) in sprejemnik (čitalec), tako kot to predstavlja slika 3. Čitalec preko radijskih valov sprejme podatke, ki mu jih posreduje oddajnik značke. Komunikacija med značko in čitalcem se vzpostavi v trenutku, ko se značka znajde v elektromagnetnem polju antene. Tako značka dobi dovolj energije za oddajo informacijskega signala. Območje dosega antene je lahko zelo različno, največji vpliv na doseg ima frekvenca, na kateri oddaja čitalec. Sistemi delujejo pri nizkih, visokih in ultra visokih frekvencah. Tabela 1 prikazuje značilna frekvenčna področja uporabe sistemov RFID, domete ter mogoče implementacije [1, 19].



Slika 3: Delovanje pasivnega sistema RFID.

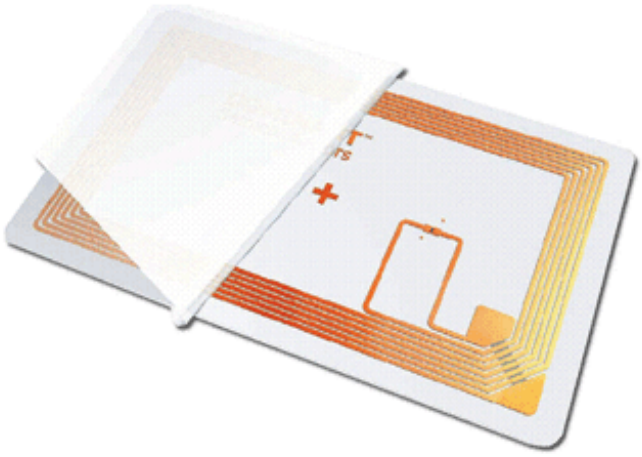
Frekvenčno področje	**Tipične frekvence**	**Domet branja**	**Aplikacije**
Nizke frekvence (LF)	125 kHz	Nekaj milimetrov ali fizični kontakt	Nadzor dostopa in plačevanje. Ponavadi se uporablja za označevanje predmetov.
Visoke frekvence (HF)	13.56 MHz	Nekaj 10 cm, do 1 m z dobrim načrtovanjem.	Predmeti „na polici“: farmacevtska, prehrambena industrija, knjižnice.
Ultra visoke frekvence (UHF)	868 MHz (Evropa), 915 MHz (ZDA)	Več metrov, v idealnem okolju preko 10 m.	Skladišča, proizvodna veriga (tekoči trak), nadzor vozil.

Tabela 1: Pregled frekvenčnih področij uporabe tehnologije RFID.

Vse frekvence so v tako imenovanem območju kratkega dosega (SRD), zato za delovanje takšnih naprav ni treba imeti posebnega dovoljenja. Območje branja sistemov pa se giblje od treh centimetrov do šestih metrov.

Značka RFID je lahko najrazličnejših oblik in dimenzij. Lahko je kot nalepka pritrjena na objekt, lahko je zaprta v stekleno kapsulo ali plastificirana v velikosti bančne kartice.

Slednje že uporabljamo pri odpiranju vrat poslovnih prostorov namesto ključa. Prikazuje jo slika 4. Za razliko od sistemov iz šestdesetih let je zdaj na tovrstne značke mogoče zapisati precej več podatkov, od 8 do 1024 bajtov.



Slika 4: Plastificirana značka RFID.

3 Analiza in načrtovanje sistema

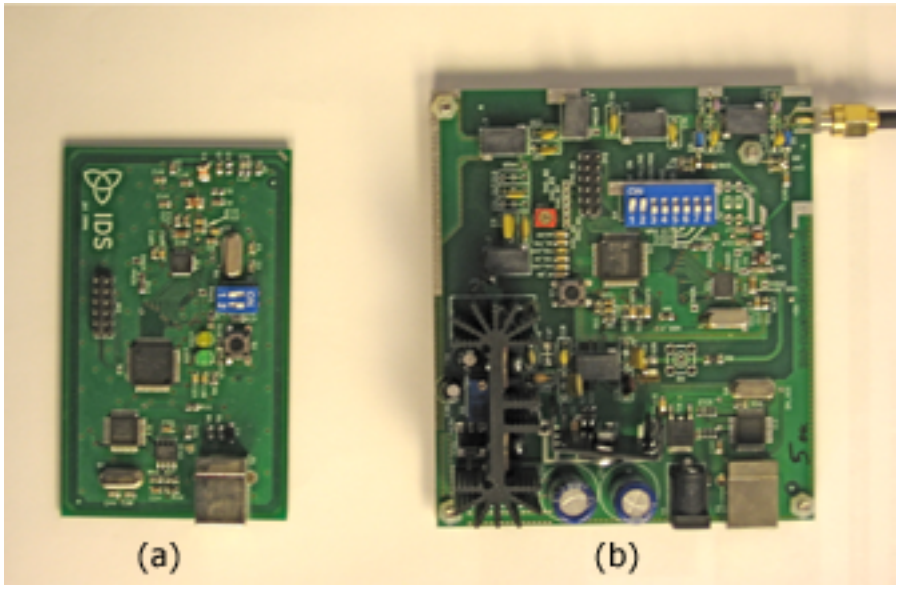
V fazi analize smo zastavili funkcionalne zahteve, ki naj jih sistem inteligentnega hladilnika podpira. Izbrali in analizirali smo strojno opremo ter njene zmogljivosti. Na podlagi ugotovljenega pa smo naredili sam načrt sistema.

3.1 Strojna oprema

Osnova strojnega dela sistema so osebni računalnik z zaslonom, občutljivim na dotik, dva čitalca RFID ter digitalna tehtnica. Dodatna oprema so še pasivne značke RFID v obliki plastificiranih kartic in nalepk.

3.1.1 Čitalca RFID

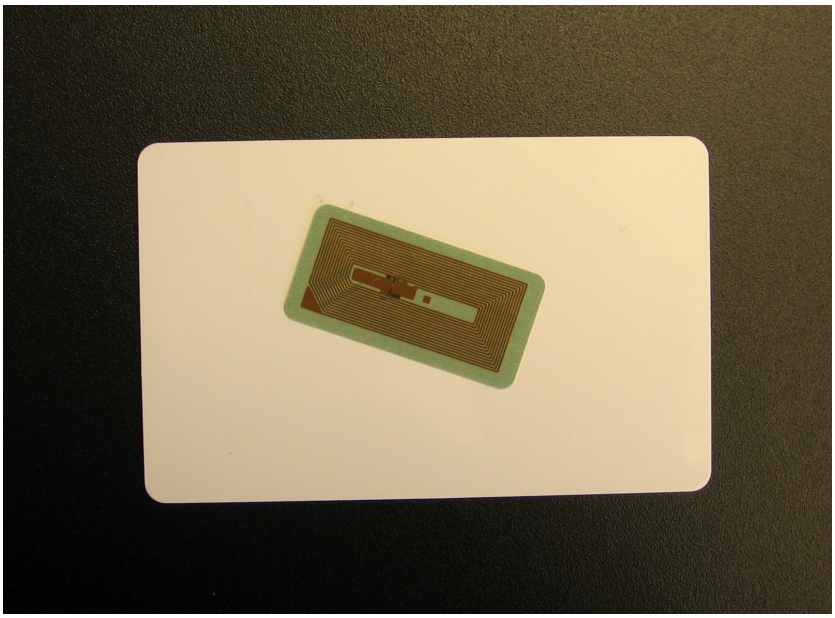
Na voljo sta dva čitalca RFID. Prvi nosi oznako IDS-R13MP in ga proizvaja podjetje IDS Microchip. Je krajšega dosega (do približno 10 cm). Drugi čitalec je izdelan v podjetju Texas Instruments, vendar pa je prav tako kot prvi razvit v podjetju IDS Microchip. Njegova oznaka je TRF7960, je močnejši in ima doseg branja do polovice metra. Oba čitalca sta prikazana na sliki 5, delujeta v frekvenčnem območju 13.56 MHz in omogočata tako branje kot tudi zapisovanje podatkov. Komunikacija z značkami RFID poteka po standardu ISO 15693. Na računalnik pa jih lahko priključimo preko vodila USB [2, 3, 5].



Slika 5: a) Čitalec IDS-R13MP. b) Čitalec TRF7960.

3.1.2 Značke RFID

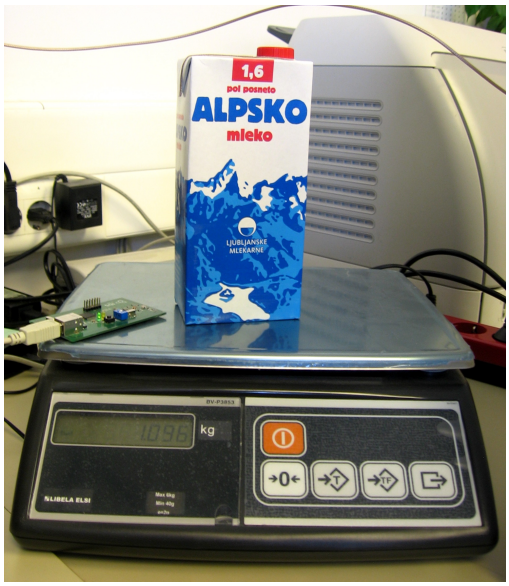
Značke RFID so od proizvajalca Philips, njihova točna oznaka pa je Standard Label IC SL2 ICS20. Nekatere so plastificirane in zavzemajo obliko bančne kartice, druge pa so nalepke. Oba tipa prikazuje slika 6. Značke delujejo v frekvenčnem območju 13.56 MHz in hkrati podpirajo standard ISO 15693 ter so popolnoma kompatibilne z zgoraj omenjenima čitalcema. Hranijo lahko do 1024 bitov podatkov, v pomnilniku tipa EEPROM, organiziranih v 32 blokov po 32 bitov. V prvih dveh blokih je zapisan enoumni identifikator (UID), ki je tovarniško določen in se ga ne da spreminjati. Sledita dva bloka podatkov za interno uporabo. Preostali bloki pa so namenjeni uporabniškim podatkom, ki jih lahko beremo in zapisujemo. Ti podatki se v pomnilniku ohranijo najmanj 10 let, mogoče pa jih je prepisati vsaj 100.000 krat [4].



Slika 6: Znački RFID. Spodaj je bela plastificirana značka, nanjo pa je položena še značka v obliki nalepke.

3.1.3 Digitalna tehtnica

Digitalna tehtnica je španskega proizvajalca DIBAL. Njena oznaka je E – 10, prikazuje jo slika 7. Omogoča komunikacijo preko RS-232 vodila.



Slika 7: Tehtnica DIBAL E-10.

3.2 Funkcionalnost

Zastavljene funkcionalnosti sistema inteligentnega hladilnika lahko združimo v tri sklope; tako so tudi predstavljene v nadaljevanju.

3.2.1 Avtomatsko vodenje vsebine hladilnika

Podatki o vsebini hladilnika naj se avtomatsko ažurirajo. Sistem naj torej neprestano periodično bere stanje preko čitalcev RFID in tako vodi evidenco vseh artiklov, njihove količine ter datum preteka uporabnosti.

3.2.2 Prikaz vsebine, obvestil in predlogov

Na podlagi evidence artiklov naj se prikaže vsebino hladilnika ob pritisku ustreznega gumba na zaslonu. Poleg tega naj nas sistem obvešča o pokvarjenih in skoraj pokvarjenih izdelkih ter izdelkih z malo vsebine. Omogoča naj se generiranje in izpis nakupovalnega seznama. Poleg tega pa naj nam inteligentni hladilnik tudi izpiše predloge posameznih obrokov, upoštevajoč vsebino, ki jo imamo.

3.2.3 Oddaljeni pregledi

Vsebino hladilnika, predloge posameznih obrokov ter vsebino nakupovalnega listka naj bo

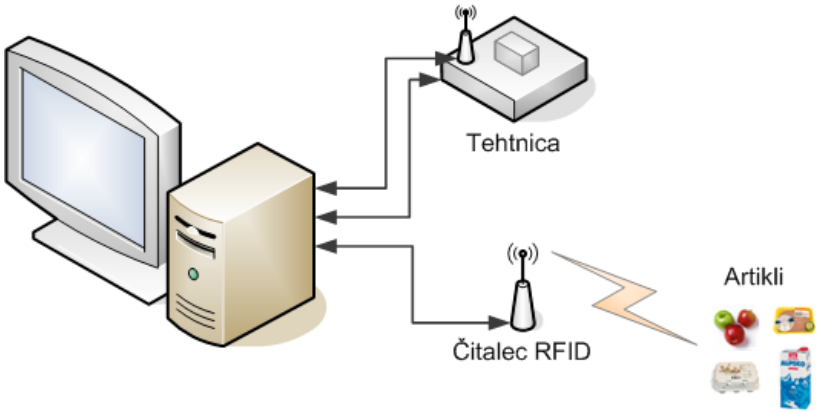
mogoče pregledati od daleč preko interneta z uporabo spletnega brskalnika. Poleg tega naj se omogoči pošiljanje nakupovalnega seznama preko elektronske pošte. Zasnova naj upošteva tudi prihodnje razširitve v smeri elektronskega nakupovanja.

3.3 Načrtovanje sistema

Na podlagi funkcijskih zahtev in uporabljene strojne opreme lahko preidemo k načrtovanju sistema.

3.3.1 Postavitev strojne opreme

Osnova strojnega dela sistema, ki je prikazan na sliki 8, je osebni računalnik z zaslonom, občutljivim na dotik. Zaslon bo poleg prikazovanja služil tudi za upravljanje z grafičnim vmesnikom aplikacije inteligentnega hladilnika. Na računalnik sta preko vodila USB priključena še dva čitalca RFID ter preko vmesnika RS-232 digitalna tehtnica. Prvi čitalec s krajšim dosegom je nameščen poleg digitalne tehtnice. Služi za detekcijo artikla na tehtnici in za posodabljanje teže, ki se sicer prebere s tehtnice, na znački artikla. Drugi čitalec pa se uporablja za detekcijo in branje podatkov vseh artiklov v inteligentnem hladilniku.



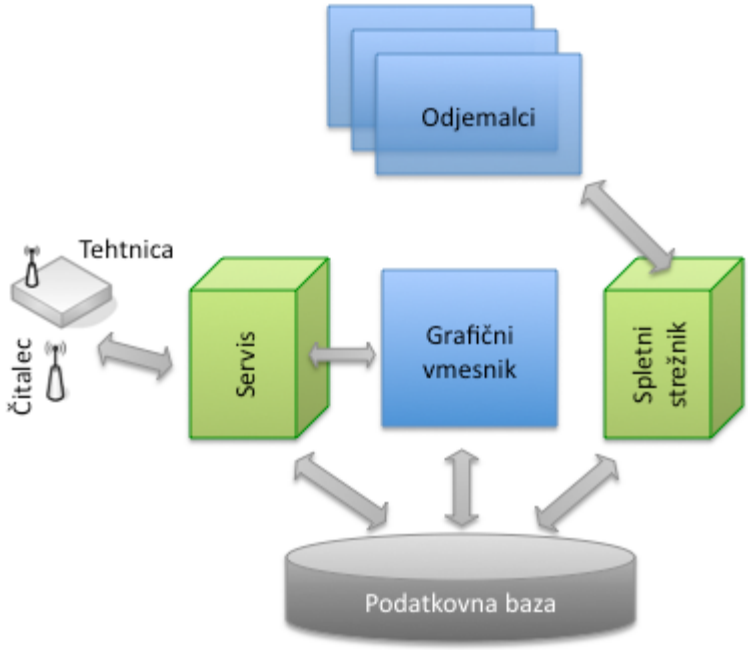
Slika 8: Strojna postavitev sistema.

3.3.2 Programska zasnova

Zasnovo programskega dela sistema inteligentnega hladilnika prikazuje slika 9. Iz nje je

razvidno, da je sistem sestavljen iz štirih ključnih delov: podatkovne baze, servisne aplikacije, aplikacije z grafičnim vmesnikom ter spletne aplikacije.

Centralni del sistema predstavlja podatkovna baza. V njej je shranjena vsebina iz prebranih značk RFID na izdelkih, ki so v hladilniku, poleg tega pa še recepti, podatki o artiklih in zgodovina o zalogah, nakupih in izbranih jedilnikih.



Slika 9: Programska shema.

Druga komponenta sistema je servis, ki periodično komunicira s čitalcema RFID in tehtnico. Iz prvega čitalca razbere, kakšna je vsebina hladilnika, in to stanje zapiše v podatkovno bazo. Z drugim pa ob detekciji artikla na tehtnici prebere njegovo težo ter ta podatek zapiše na značko artikla. V primeru, da je aktiviran grafični vmesnik, pa servis komunicira še z njim in mu sporoča, kateri artikel je na tehtnici ter kdaj se spremeni vsebina hladilnika, tako da grafični vmesnik ustrezno posodobi svojo zaslonsko masko. Slednja funkcionalnost ni ključnega pomena, vendar pa bistveno pripomore k zmogljivosti sistema, saj bi v nasprotnem primeru morala aplikacija z grafičnim vmesnikom neprestano

poizvedovati v podatkovni bazi, da bi ugotovila spremembo v zalogi in jo prikazala.

Naslednji ključni del je aplikacija z grafičnim vmesnikom. Grafični vmesnik je prirejen za delo z zaslonom, občutljivim na dotik, in je namenjen osnovni interakciji uporabnika s sistemom inteligentnega hladilnika. Funkcije grafičnega vmesnika omogočajo prikaz vsebine hladilnika, izpis predlogov posameznih obrokov glede na vsebino hladilnika, grafični prikaz artikla, postavljenega na tehtnico, in spreminjanje le-tega ter navsezadnje prikaz nakupovalnega seznama, ki ga je mogoče posredovati po elektronski pošti.

Zadnja komponenta sistema pa je spletna aplikacija, ki se izvaja na spletnem strežniku. Omogoča, da se podatki, ki se sicer prikazujejo na grafičnem vmesniku lokalne aplikacije, prikažejo še na oddaljenih sistemih oziroma napravah. Tako lahko preko spletne aplikacije na oddaljenih sistemih pregledujemo trenutno zalogo, predloge posameznih obrokov, upoštevajoč zalogo, ter vsebino nakupovalnega listka.

4 Implementacija

Pred samo implementacijo sem pregledal trenutno aktualna razvojna okolja, same tehnologije, kaj nam omogočajo, in na podlagi ugotovljenega ter lastnih prioritet izbral tehnologijo in razvojna okolja, v katerih sem sistem implementiral.

4.1 Pregled tehnologij

Že dalj časa so popularne tehnologije, ki omogočajo izdelavo uporabniških aplikacij tako, da delujejo na različnih platformah. Večja potreba po takšnih aplikacijah se je pojavila prav v zadnjem času, ko se najbolj razširjena mikroprocesorska arhitektura x86 (32-bitna) poslavlja in jo nadomešča nova x64 (64-bitna). Praktično vsi operacijski sistemi, ki so prešli iz arhitekture x86 na x64, še vedno podpirajo izvajanje starih 32-bitnih aplikacij, vendar je za to potrebna emulacija in delovanje takih aplikacij ni optimalno. Nove 64-bitne aplikacije pa na starejših verzijah teh operacijskih sistemov sploh ne delujejo. Rešitev tega so večplatformske aplikacije. Te se ne prevedejo za specifično platformo, ampak se ob izvajanju interpretirajo. Odvisno od uporabljene tehnologije se nekatere interpretirajo neposredno iz izvorne kode, druge pa iz vmesne kode (byte code). Tipična predstavnika prvega načina sta jezika PHP in Perl, drugega pa tehnologiji Java in Microsoft .NET. Slednja tehnologija omogoča izdelavo aplikacij, ki enako dobro delujejo na 32- in 64-bitnih različicah operacijskih sistemov Microsoft Windows. Izvajanja takih aplikacij na drugih operacijskih sistemih Microsoft ne podpira, vendar pa je vseeno mogoče z uporabo odprtokodnega izvajalnega ogrodja Mono. Tehnologija Java pa tako kot PHP in Perl že v osnovi deluje na praktično vseh bolj razširjenih arhitekturah in operacijskih sistemih [14, 20, 22].

Kljub temu razvoj aplikacij v prevajalskih programskih jezikih, kot so: C, C++, Visual C++, Objective-C, Object Pascal itn., ne pojenja. Glavna prednost prevedenih aplikacij je v njihovi hitrosti izvajanja, ki je lahko tudi deset- in večkrat hitrejša od interpreterskih. Poleg tega prenos takih aplikacij iz 32-bitnih verzij na 64-bitne ni prav posebej težak, večinoma jih je treba le ponovno prevesti.

V našem primeru je treba razviti tri ločene aplikacije, ki se med seboj prepletajo in dopolnjujejo.

- Servis je aplikacija, ki naj teče v ozadju tako, da ni pod neposredno kontrolo uporabnika. Take aplikacije se na operacijskih sistemih Microsoft Windows

označujejo kot Windows Service oziroma je na starejših različicah v uporabi oznaka NT Service, na praktično vseh ostalih operacijskih sistemih pa se uporablja oznaka Daemon. Implementacija takih servisov je mogoča z uporabo sledečih tehnologij [13]:

- Windows Service: Samo Microsoft .NET
- NT Service: Visual C++, Object Pascal, Visual Basic, itn.
- Daemon: C, C++, Perl, itn.

- Aplikacijo z grafičnim vmesnikom je mogoče razviti praktično v vsakem programskem jeziku. Za gradnjo uporabniških grafičnih vmesnikov (GUI) se uporabljajo knjižnice, ki za posamezen operacijski sistem bolj ali manj poenotijo izgled takih aplikacij. Te so:

  - Microsoft Windows: Windows API
  - Mac OS X: Cocoa in Carbon
  - GNU/Linux in ostali Unix sistemi: GTK+, Qt, Motif itn.

  Za gradnjo večplatformskih grafičnih vmesnikov sta v tehnologiji Java na voljo dve knjižnici, AWT in Swing, ki poenotita izgled Java aplikacij na vseh platformah hkrati. Ti knjižnici sta neodvisni od operacijskega sistema, zato se tudi izgled takih grafičnih vmesnikov razlikuje od ostalih platformsko specifičnih na dotičnem sistemu. Drug način za gradnjo večplatformskih grafičnih vmesnikov je uporaba nove knjižnice, ki predstavlja vmesno plast med našo aplikacijo in grafičnimi knjižnicami posameznega operacijskega sistema. Primer tega sta knjižnici SWT (Java) in Windows Forms (Microsoft .NET in Mono). Z uporabo takih knjižnic dosežemo, da večplatformski grafični vmesnik na operacijskem sistemu izgleda naravno ostalim platformsko specifičnim [21].

- Za razvoj spletne aplikacije potrebujemo tehnologijo, ki nam omogoča dinamično generiranje spletnih strani. Na voljo imamo več možnosti, najpopularnejše pa so: ASP .NET, PHP in JSP. Vse tri delujejo na številnih platformah (ASP .NET pod ogrodjem Mono) in omogočajo hiter razvoj aplikacij (RAD).

4.2 Izbira tehnologije

Glavno vodilo pri izbiri tehnologije je bila želja, da bi bil zamišljeni sistem inteligentnega hladilnika čim splošnejši, da bi deloval na čim večjem številu različnih računalnikov in posledično na različnih operacijskih sistemih. Poleg tega sem dajal prednost odprtokodnim rešitvam.

4.2.1 Lastne aplikacije

Za razvoj lastnih komponent sistema sem se odločil uporabiti Javansko tehnologijo, ki omogoča, da ista izvršilna datoteka deluje na vseh podprtih operacijskih sistemih. Vendar pa tukaj vseeno nastopi težava pri komunikaciji s strojno opremo. Ta se namreč razlikuje od sistema do sistema. V našem primeru imamo opravka z napravami, priključenimi na vodilo RS-232 in vodilo USB. Čitalca RFID, priključena na vodilo USB, emulirata vmesnik RS-232, tako da tudi do teh dveh naprav s programerskega stališča dostopamo enako kot do tehtnice, ki je tudi fizično priključena na vrata RS-232. Komunikacija z vodilom RS-232 je na različnih operacijskih sistemih precej podobna, razlike so le v naslavljanju vrat. Kljub temu pa sem se z ozirom na možnost priklapljanja vedno novih naprav, dejstva, da so programi v Javi dokaj omejeni pri neposredni komunikaciji z napravami, priključenimi na različna vodila, ter navsezadnje na nezmožnost poganjanja Javanskih aplikacij v Windows Service načinu odločil, da bo servisni del sistema platformsko specifičen in realiziran v drugi tehnologiji.

Tako sta grafični vmesnik in spletna aplikacija implementirana z uporabo Javanske tehnologije, servis za komunikacijo s strojno opremo pa v Microsoftovi tehnologiji .NET [14, 22].

4.2.2 Baza podatkov

Za hrambo podatkov sem izbral odprtokodno relacijsko podatkovno bazo PostgreSQL. Baza je na voljo za praktično vse splošno razširjene operacijske sisteme, tako zadosti obema kriterijema izbire. Poleg tega je PostgreSQL po besedah avtorjev najnaprednejša odprtokodna podatkovna baza, kar pa lahko glede na pretekle izkušnje z delom na tem področju potrdim tudi sam.

Za delo s podatkovno bazo se uporabljajo standardizirani stavki SQL, tako je baza tudi preprosto nadomestljiva s kakšno drugo, kot so: MSSQL, DB2, Oracle, MySQL, SQLite itn. Vse, kar je treba v tem primeru storiti, je zamenjava knjižnice za dostop do baze, vsa logika pa ostane ista.

4.2.3 Spletni aplikacijski strežnik

Za serviranje spletnih strani sem prav tako izbral zelo razširjen odprtokodni večplatformski spletni aplikacijski strežnik Apache Tomcat [10].

4.3 Razvojna orodja

Razvoj servisa .NET sem opravil v razvojnem okolju Microsoft Visual Studio 2008. Aplikacijo z grafičnim vmesnikom ter spletno aplikacijo sem razvil v okolju Eclipse JEE Ganymede. Poleg tega sem pri razvoju slednje za izdelavo posameznih grafičnih elementov uporabil tudi orodje za manipulacijo z slikami Adobe Photoshop.

4.4 Servis

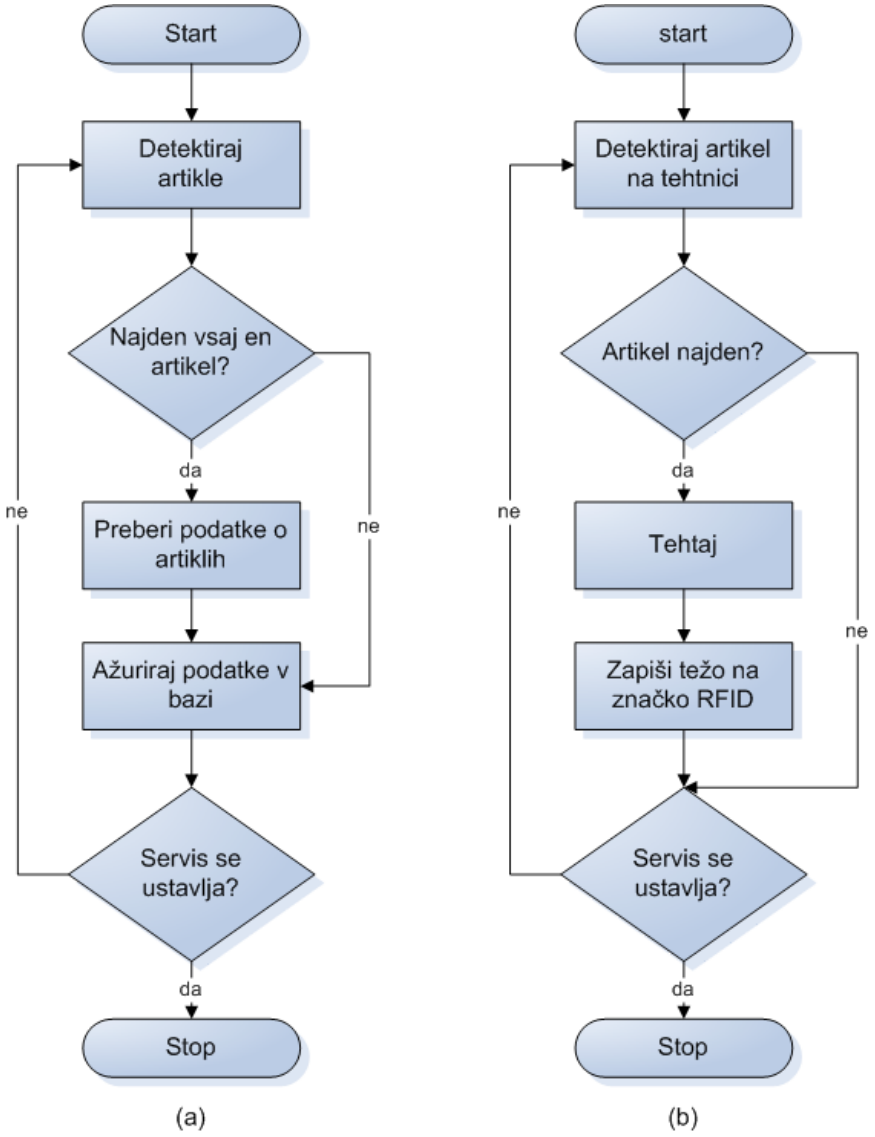
Servisna aplikacija je razvita za operacijski sistem Microsoft Windows. Napisana je v jeziku C# v načinu, da deluje kot Windows Service. Prednost takega načina izvajanja je v tem, da lahko aplikacija vedno deluje v ozadju, neodvisno od uporabniškega profila trenutno prijavljenega uporabnika na operacijskem sistemu. Slednja lastnost je tudi nujno potrebna, saj mora servis preko zunanjih naprav neprestano preverjati trenutno stanje zaloge v hladilniku in to zapisovati v bazo podatkov.

Kot je bilo že omenjeno, je servis zadolžen za več ločenih opravil. Te lahko smiselno združimo v tri ločene skupine:

- Detekcija artiklov v hladilniku, branje njihovih podatkov ter zapisovanje le-teh v podatkovno bazo. Za detekcijo in branje podatkov artikla se uporabi čitalec RFID na sliki 5 b). Diagram te procedure prikazuje slika 10 a).

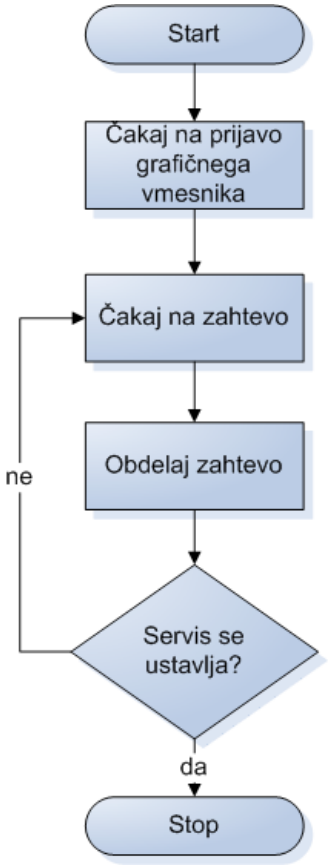
- Detekcija artikla na tehtnici, tehtanje ter ponovni zapis podatkov v njegovo značko RFID. Tukaj pa se za detekcijo, branje in zapisovanje podatkov artikla v njegovo značko RFID uporablja čitalec RFID na sliki 5 a). Diagram te procedure pa je prikazan na sliki 10 b).

- Komunikacija z aplikacijo z grafičnim vmesnikom je prikazana na sliki 11. Servis ji sporoča, kdaj se spremeni zaloga in kateri artikel je na tehtnici ter njegovo težo. V grafičnem vmesniku pa imamo možnost spreminjanja razpoznanega artikla, tako da to spremembo potem tudi ustrezno posreduje servisu, ki jo s strojno opremo tudi uveljavi.



Slika 10: a) Prikazana je osnovna procedura za ažuriranje vsebine hladilnika. b) Procedura za tehtanje posameznega artikla, postavljenega na tehtnico.

Vsaka od treh zgoraj naštetih skupin ima vsaj eno operacijo, ki zaradi čakanja na odziv ostalih komponent traja veliko časa (velikostnega reda nekaj 100 ms). Pri prvih dveh skupinah je to čakanje na odgovor strojne opreme, pri komunikaciji pa se najprej čaka, da se grafični vmesnik sploh požene, nadalje pa se čaka na njegove zahtevke oziroma povpraševanja. Komunikacija namreč deluje v načinu odjemalec-strežnik, kjer je servis strežnik, aplikacija z grafičnim vmesnikom pa odjemalec, ki periodično izprašuje.



Slika 11: Diagram procedure za komunikacijo z grafičnim vmesnikom.

Zato da se časi čakanja posameznih funkcijsko ločenih skupin operacij ne bi med seboj seštevali, je smiselno le-te izvajati paralelno. To je vsaj v primeru komunikacije z grafičnim vmesnikom tudi nujno, saj bi v nasprotnem primeru v času, ko aplikacija z grafičnim vmesnikom ne bi tekla, celoten servis stal in čakal na ponoven zagon grafičnega vmesnika, namesto da bi ostale operacije nemoteno izvajal dalje.

Vse tri skupine operacij, ki so prikazane na slikah 10 in 11, so implementirane tako, da se izvajajo znotraj enega procesa, ampak vsaka v svoji niti. To pa prinese nove težave, saj se lahko zgodi, da dve ali celo vse tri niti istočasno dostopajo do istega skupnega vira. Rezultat tega pa je nepredvidljiv, v najslabšem primeru privede to izgube podatkov. Problem sem rešil tako, da sem vse skupne spremenljivke postavil znotraj enega objekta. Niti dostopajo do teh spremenljivk preko javnih metod objekta. V teh metodah pa je poskrbljeno, da se najprej zaklene dostop do samega objekta, nato se izvedejo same operacije nad skupnimi spremenljivkami, nazadnje pa se dostop do objekta tudi ponovno

odklene. S tem je zagotovljeno, da se operacije znotraj takih metod izvedejo serijsko od začetka do konca, ne da bi jih vmes prekinila kakšna druga nit. Za boljšo ilustracijo je na sliki 12 prikazan izsek take kode. Znotraj objekta „CommData“ je metoda „getRefresh“, ki prebere in nastavi spremenljivko „refresh“ z uporabo zgoraj opisanega zaklepanja objektov.

```
public class CommData
{

    bool refresh = true;

    ...

    public CommData()
    {
    }

    public bool getRefresh()
    {
        bool res = false;

        lock (this)
        {
            res = refresh;
            refresh = false;
        }
        return res;
    }

    ...

}
```

Slika 12: Način sinhronizacije med nitmi z zaklepanjem virov.

4.4.1 Komunikacija s strojno opremo

S programerskega stališča komunikacija s strojno opremo v našem primeru pomeni le pisanje posameznih ukazov ter branje rezultatov le-teh iz vrat RS-232 v predpisanih časovnih intervalih.

4.4.1.1 Digitalna tehtnica

Digitalna tehtnica je preprostejša naprava in za delovanje ne potrebuje inicializacije, saj deluje le v enem načinu. Zahtevek za podatek o teži podamo tako, da na serijska vrata

zapišemo ukaz v naslednjem formatu: `98PPPPPCCrLf`.

Vsi znaki v tem formatu so kodirani po kodni tabeli ASCII. Pomen posameznih znakov pa je naslednji:

- `98`: Konstanta, ki predstavlja identifikator ukaza.
- `PPPPP`: Petmestna desetiška številka, namenjena podatku o ceni artikla. Tega v našem primeru ne bomo potrebovali, zato na to mesto vedno vpišemo pet ničel.
- `C`: Paritetni znak za preverjanje pravilnosti prenosa ukaza med računalnikom in tehtnico. Izračuna se ga z ekskluzivno disjunkcijo (XOR) nad vsemi znaki do znaka C.
- `CrLf`: Konstanta oziroma znak za novo vrstico, ki označuje konec ukaza.

Tehtnica pa na zgornji ukaz odgovori v formatu: `99SWWWWWE I I I I I I CCrLf`.

Iz tega lahko razberemo podatek o teži artikla na tehtnici. Pomen posameznih znakov je naslednji:

- `99`: Konstanta, ki predstavlja identifikator odgovora na ukaz.
- `S`: Status za maso. Ko je znak S enak 1, pomeni, da masa ni v mirovanju, torej je neveljavna. Ko pa je ta znak enak 0, dobimo v naslednjih znakih podatek o pravi teži artikla.
- `WWWWW`: Petmestna desetiška številka, ki predstavlja težo artikla, postavljenega na tehtnico.
- `E`: Status za znesek. Ko je E enak 1, je le-ta neveljaven, ko pa je enak 0, je v redu. Ker podatka o znesku ne potrebujemo, se na ta znak ne bomo ozirali.
- `I I I I I I`: Šestmestna desetiška številka, ki predstavlja izračunan znesek glede na podano ceno in težo. Tudi tega podatka v našem primeru ne potrebujemo.

- `C`: Paritetni znak za preverjanje pravilnosti prenosa odziva na ukaz med tehtnico in računalnikom. Izračuna se ga z ekskluzivno disjunkcijo (XOR) nad vsemi znaki do znaka C.

- `CrLf`: Znak za novo vrstico, ki označuje konec podatkov.

4.4.1.2 Čitalec RFID

Čitalca RFID sta bistveno bolj zapletena, podpirata množico operacij, ki jih lahko izvajata na več različnih načinov. Kompatibilna sta s ISO 15693 standardom, ta pa tudi določa sam nabor funkcij in načine delovanja. Čitalca imata poleg ukazov ISO 15693 standarda tudi nekaj svojih, ti pa so v takem formatu, da so v skladu s standardom. To je mogoče zato, ker so bile take razširitve v standardu predvidene. V nadaljevanju bodo zaradi preobsežnosti standarda in njegovih razširitev predstavljeni samo ukazi, ki so bili tudi uporabljeni. Način delovanja čitalca RFID določimo s posebnimi ukazi ob inicializaciji naprave. Ti ukazi so:

- Set Full RF Power: S tem ukazom določimo, ali naj antena čitalca deluje s polno (200 mW) ali polovično (100 mW) močjo. Pri delovanju s polno močjo so zmogljivosti seveda boljše.

- Register Write: V register naprave se zapišejo zastavice, ki označujejo nekatere lastnosti pri komunikaciji z oddajniki. Uporabljeni sta bili dve zastavici:

    - Subcarrier: Tukaj izberemo uporabo enega ali dveh RF podnosilnikov.

    - Hight Data Rate: S tem izberemo hitri ali počasnejši prenos podatkov. Pri hitrejšem prenosu komunikacija poteka hitreje, vendar na račun dometa.

- Set AGC: S tem ukazom vključimo ali izključimo avtomatsko kontrolo ojačanja. Vklop te funkcionalnosti je priporočljiv predvsem v okoljih z veliko šuma.

- Set Receiver Mode: Tukaj lahko izbiramo med kanaloma AM in PM. AM je privzeti, PM pa pomožni. Na pomožni kanal preklopimo, kadar je kakovost sprejema na osnovnem kanalu slabša od sprejema na kanalu PM.

Ko je naprava enkrat inicializirana, lahko izvajamo ostale ukaze za samo detekcijo značk RFID, za njihovo branje in pisanje. Uporabljeni so bili sledeči ukazi:

- Inventory: Na ta ukaz čitalec RFID odgovori s seznamom vseh značk v svojem dosegu. Značke RFID so v seznamu predstavljene z 64-bitnim enoumnim identifikatorjem UID.

- Read single block: S tem ukazom iz značke RFID preberemo posamezen 32-bitni blok podatkov. Številka bloka in zastavice, ki označujejo podrobnosti pri prenosu podatkov podobno kot pri ukazu Register Write, so obvezni parametri. Parameter z identifikatorjem značke pa je opcijski, vendar je v našem primeru vedno uporabljen, saj nenaslovljeno branje deluje samo v primeru, ko je v območju dosega samo ena značka RFID.

- Write single block: Ta ukaz deluje popolnoma enako kot ukaz Read single block, le da se namesto branja podatkov izvrši pisanje.

Tudi pri komunikaciji s čitalcema RFID so vsi ukazi in odgovori na ukaze predstavljeni z ASCII znaki, ki potujejo po vodilu RS-232. Število vseh ukazov in njihovih inačic je preveliko, zato bo za ilustracijo prikazano samo kodiranje ukaza in odgovora na ukaz Inventory.

Primer konkretnega ukaza Inventory predstavljajo ASCII znaki:

`010B000304140601000000`.

Pomen posameznih znakov je naslednji:

- `01`: Začetek okvirja.

- `0B`: Šestnajstiško predstavljena številka dolžine paketa v bajtih. V tem primeru to pomeni dolžino enajstih bajtov.

- `00`: Konstanta.

- `0304`: Začetek vsebine paketa.

- `14`: Identifikator ukaza. V tem primeru je to identifikator ukaza Inventory.
- `06`: Zastavice. V tem primeru je postavljena samo zastavica High Data Rate, ki je že opisana pri ukazu Register Write.
- `01`: Izogibanje koliziji.
- `00`: Konstanta.
- `0000`: Konec okvirja.

Odgovor na prejšnji ukaz čitalec RFID poda z naslednjim izpisom:

```
010B0003041406010000000
ISO 15693 Inventory request.
80T42[z,40]
01N[,40]
01N[,40]
01N[,40]
01N[,40]
01N[,40]
01N[,40]
01N[,40]
01N[,40]
01N[,40]
01N[,40]
60F40E[1B298C15000104E0,6D]
01N[,40]
01N[,40]
01N[,40]
01N[,40]
>
```

V prvi vrstici izpisa je vrnjeno zaporedje znakov, ki smo ga poslali napravi (Inventory). V drugi vrstici nas naprava obvešča, da je ta ukaz definiran v standardu ISO 15693. V naslednjih šestnajstih vrsticah pa so dejansko uporabni podatki. Vsaka taka vrstica predstavlja časovno režo v povpraševalnem ciklu ukaza Inventory, v kateri posamezna značka RFID poda svoj odgovor. V primeru, da dve znački odgovorita v isti časovni rezini, pride do kolizije. Glede na to, da je časovnih rezin končno število 16, pride do kolizije vsaj v eni časovni rezini takrat, ko je v dosegu čitalca 17 značk RFID in več. Same postopke za odpravljanje kolizij izvaja čitalec ob naslednjem klicu ukaza Inventory, pri katerem je bila kolizija ugotovljena. V programu moramo tako poskrbeti, da se ukaz

Inventory pokliče tolikokrat, da na koncu v nobeni časovni rezini ne pride več do kolizije. Tretja vrstica v zgornji kodi predstavlja odziv značke v časovni reži številke 0, pomen posameznih znakov pa je sledeč:

- `80T42`: Status registra IRQ.
    - `80T`: Konec prenosa.
    - `42`: Kolizija.
- `[z,40]`:
    - `z`: To mesto je namenjeno identifikatorju značke, vendar pa je v tem primeru, ko je prišlo do kolizije, zapisana konstanta z.
    - `40`: Vrednost registra RSSI.

Naslednjih deset vrstic nam pove, da se v časovnih režah od 1 do 10 ni odzvala nobena značka RFID. V dvanajsti vrstici pa lahko vidimo odziv značke v časovni rezini številke 11. Pomen posameznih znakov je sledeč.

- `60F40E`: Status registra IRQ.
    - `60F`: Sprejemni medpomnilnik je 75-odstotno poln.
    - `40E`: Konec sprejema.
- `[1B298C15000104E0,6D]`:
    - `1B298C15000104E0`: Identifikator odzvane značke, zapisan v obratnem vrstnem redu.
    - `6D`: Status registra RSSI.

4.4.2 Komunikacija z grafičnim vmesnikom

Komunikacija med servisom in aplikacijo z grafičnim vmesnikom poteka preko vtičnice na vratih 1899 na način odjemalec-strežnik. V tem primeru servis deluje kot strežnik, grafični vmesnik pa kot odjemalec. Servis tako čaka na zahtevke odjemalca, jih obdela in nanje odgovori. Podprta sta dva tipa zahtevkov:

- data: Ta zahtevek je namenjen periodičnemu povpraševanju odjemalca in vrača podatke o artiklu, postavljenem na tehtnico, ter podatek o tem, ali je prišlo do spremembe zaloge v hladilniku.

- setItem: Ta zahtevek poda odjemalec takrat, ko uporabnik na grafičnem vmesniku označi, da je na tehtnico položen drug artikel, kot je to zapisano v njegovi znački RFID. Servis ob tem zahtevku zapiše posredovane spremembe v značko artikla.

4.5 Grafični vmesnik

Aplikacija z grafičnim vmesnikom je bila napisana v programskem jeziku Java. Primarno za operacijski sistem Microsoft Windows, deluje pa tudi na mnogih drugih, kot so GNU/Linux, Mac OSX, AIX itd. Grafični vmesnik je zgrajen z uporabo gradnikov odprtokodne knjižnice SWT. Za prikaz grafičnih elementov ta knjižnica nadalje uporablja knjižnice za uporabniške vmesnike operacijskega, sistema na katerem teče. Zato je izgled grafičnega vmesnika naraven izgledu ostalih programov operacijskega sistema. To je pomembno, saj uporabnikom, vajenim svojega operacijskega sistema, omogoča boljšo uporabniško izkušnjo [21].

Funkcije grafičnega vmesnika so:

- Pregled vsebine hladilnika.

- Pregled predlogov posameznih obrokov glede na vsebino hladilnika.

- Sestavljanje in pošiljanje nakupovalnega listka.

- Prikaz teže artikla, postavljenega na tehtnico.

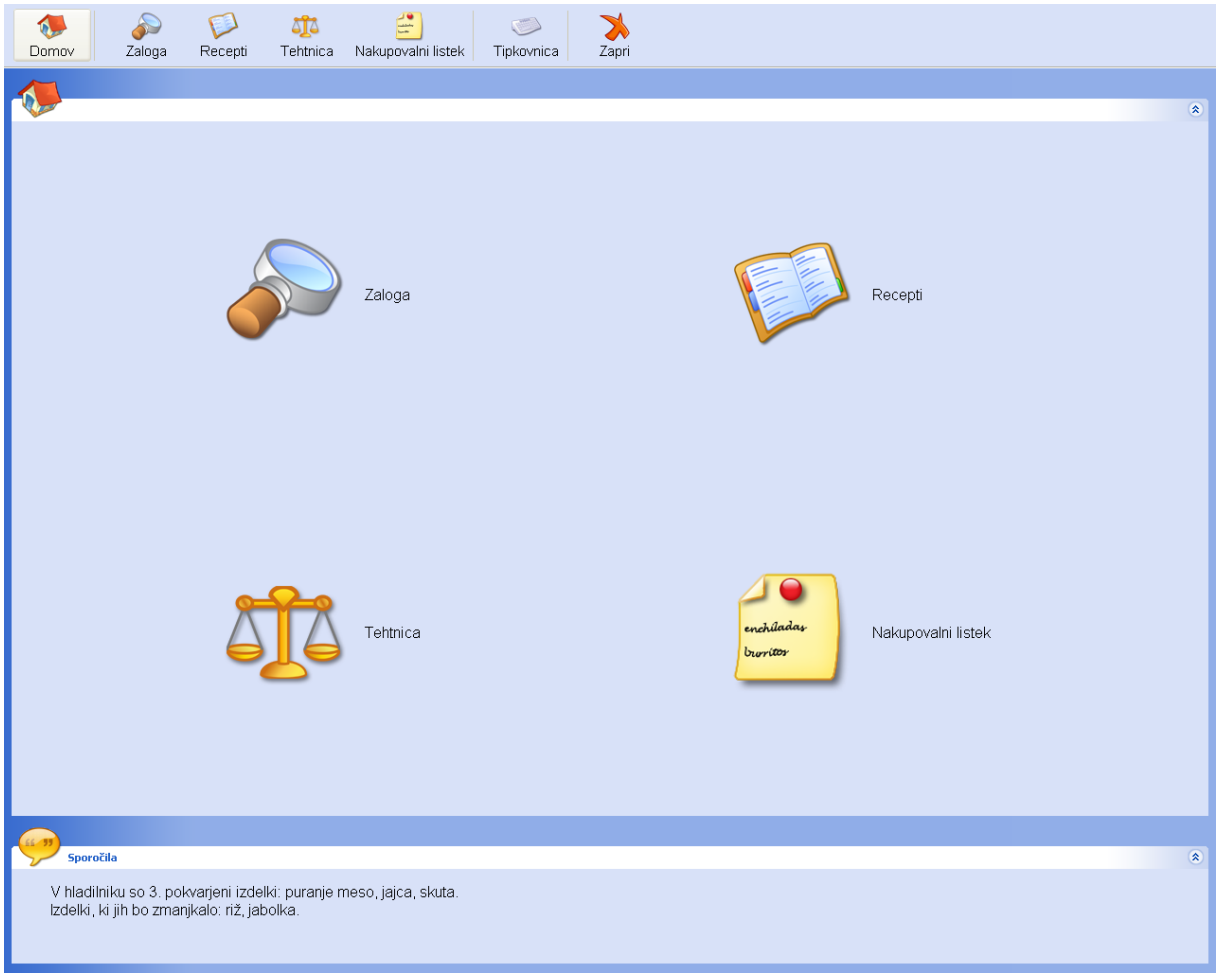
- Prikaz sporočil o pokvarjenih in skoraj pokvarjenih artiklih ter artiklih z malo vsebine.

Zgoraj naštete funkcionalnosti so v programu razporejene po posameznih zaslonskih maskah. Te maske so prilagojene za delo z zaslonom, občutljivim na dotik. Prikazane so v celozaslonskem načinu, podrobnejši opis posameznih mask sledi v naslednjih podpoglavjih.

Aplikacija z grafičnim vmesnikom komunicira tudi s servisom. Ta komunikacija je opisana v podpoglavju 4.5.6.

4.5.1 Osnovna zaslonska maska

Slika 13 prikazuje osnovno zaslonsko masko, ki se nam prikaže ob zagonu aplikacije. Vse zaslonske maske so zgrajene iz treh osnovnih gradnikov. Na vrh zaslona je postavljena orodna vrstica z gumbi. Prvih pet gumbov od leve proti desni omogoča navigacijo po posameznih zaslonskih maskah. Gumb „Tipkovnica“ nam prikaže zaslonsko tipkovnico, s katero lahko preko zaslona v posamezna polja tudi pišemo. Zadnji gumb na desni pa zapre aplikacijo.



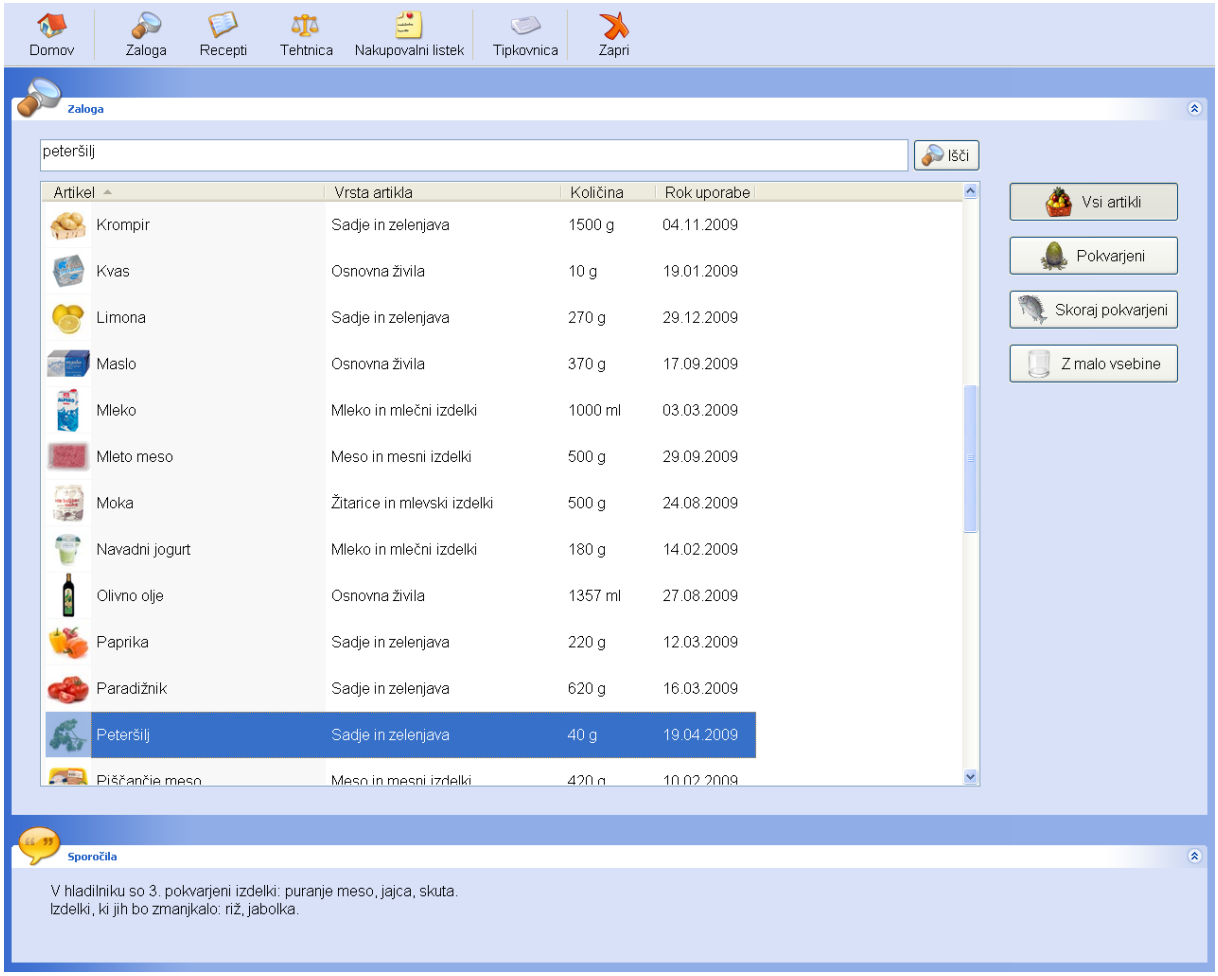
Slika 13: Osnovna zaslonska maska.

Osrednji del zaslona je namenjen posamezni funkcionalnosti, na primer: prikazu vsebine hladilnika. Tukaj imamo v tem primeru prikazan le meni, s katerim izberemo eno od preostalih zaslonskih mask. V spodnjem delu pa so prikazana sporočila, ki jih lahko skrijemo ali pa razširimo še čez osrednji del zaslona. Seveda se v tem primeru razširi ali pa

skrije tudi prejšnja vsebina osrednjega dela zaslona.

4.5.2 Zaslonska maska vsebine hladilnika

Zaslonska maska vsebine hladilnika je prikazana na sliki 14. Predstavljena je v tabeli, kjer so vpisani vsi artikli s pripadajočimi podatki. Artikle v tabeli lahko sortiramo naraščajoče ali padajoče po posameznih stolpcih oziroma po imenu, vrsti in roku uporabnosti. To storimo s pritiskom na polje z imenom posameznega stolpca na vrhu tabele.



Slika 14: Zaslonska maska zaloge hladilnika.

Artikle lahko tudi iščemo. Iskanje sprožimo s pritiskom na gumb „Išči“. Zadetki se iščejo v sortiranem stolpcu po ključni besedi, zapisani v polju nad tabelo. Na sliki 14 je prikazan primer takega iskanja s ključno besedo „peteršilj“ po imenih artiklov. Ponovno iskanje z isto ključno besedo se začne pri naslednjem artiklu od prejšnjega zadetka. Poleg tega imamo na desni strani še meni gumbov, s katerimi filtriramo artikle. S pritiskom na gumb

„Pokvarjeni“ se nam v tabeli prikažejo že pokvarjeni artikli. Gumb „Skoraj pokvarjeni“ prikaže samo artikle, ki se bodo pokvarili v naslednjem dnevu, gumb „Z malo vsebine“ prikaže le artikle z 10 % vsebine ali manj, navsezadnje pa je tu še gumb „Vsi artikli“, ki pa zbriše filter in prikaže vse artikle.

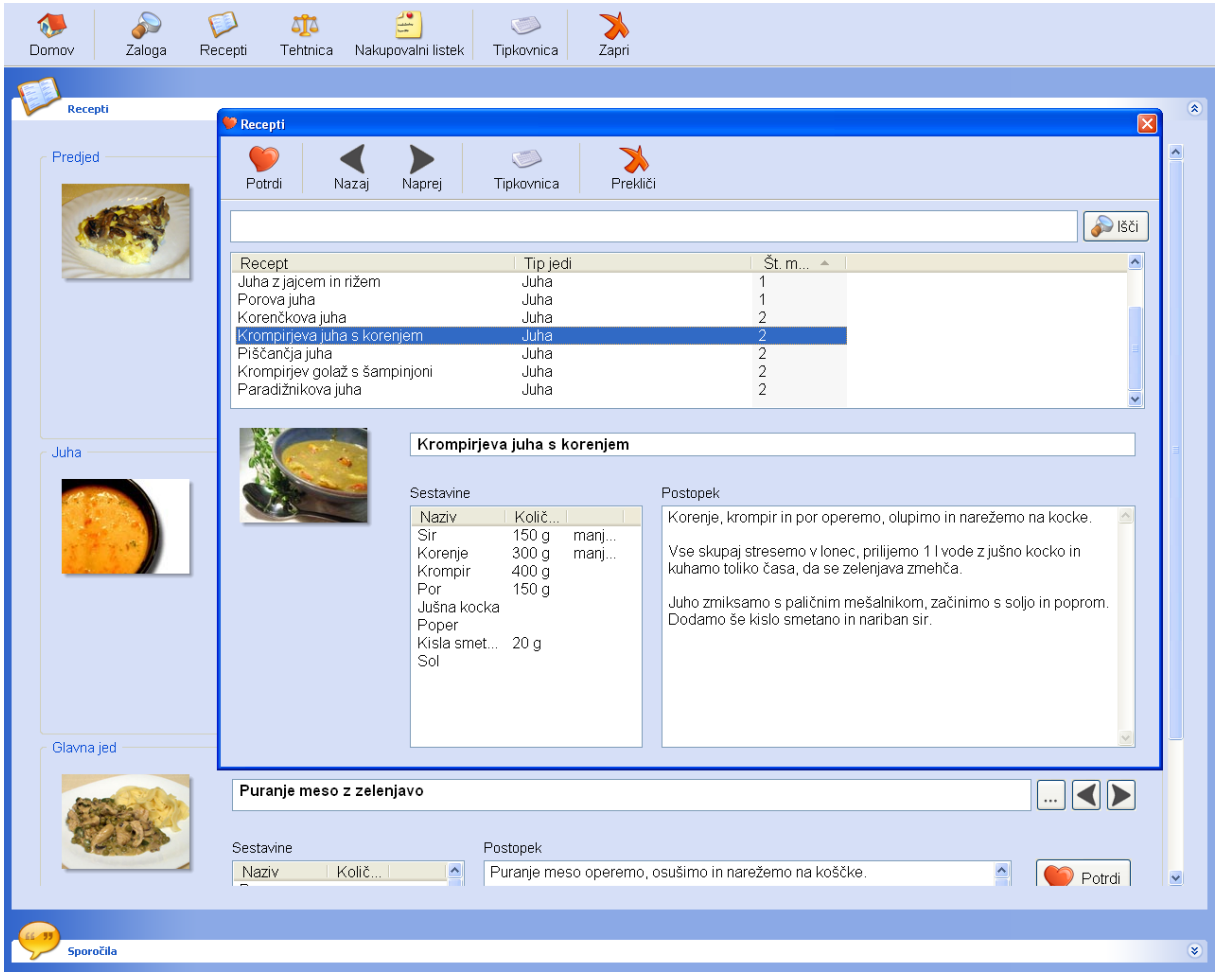
4.5.3 Zaslonska maska predlogov receptov

Slika 15 prikazuje zaslonsko masko predlogov receptov posameznih jedi glede na zalogo.

Slika 15: Zaslonska maska predlogov receptov.

Predlogi obrokov so predstavljeni po skupinah. Na vrhu so predstavljeni predlogi predjedi, nato jim sledijo predlogi juh in glavnih jedi. Na zgornji levi strani vsake take skupine je prikazana slika predlaganega obroka. Desno poleg nje je zapisano ime recepta, pod njim pa sta seznam potrebnih sestavin ter sam postopek priprave. Na skrajni desni strani zgoraj je skupina treh gumbov. Z desnima dvema se pomikamo naprej in nazaj med

predlogi. Levi pa nam odpre novo pojavno okno, kjer lahko izbiramo med vsemi recepti, ne samo med predlogi. Pod temi gumbi sta še gumba „Potrdi“, s katerim potrdimo izbiro predloga, ter gumb „Listek“, ki v primeru, da je viden, označuje, da nam v zalogi manjka določena sestavina. S pritiskom na ta gumb dodamo manjkajoče sestavine na nakupovalni listek.



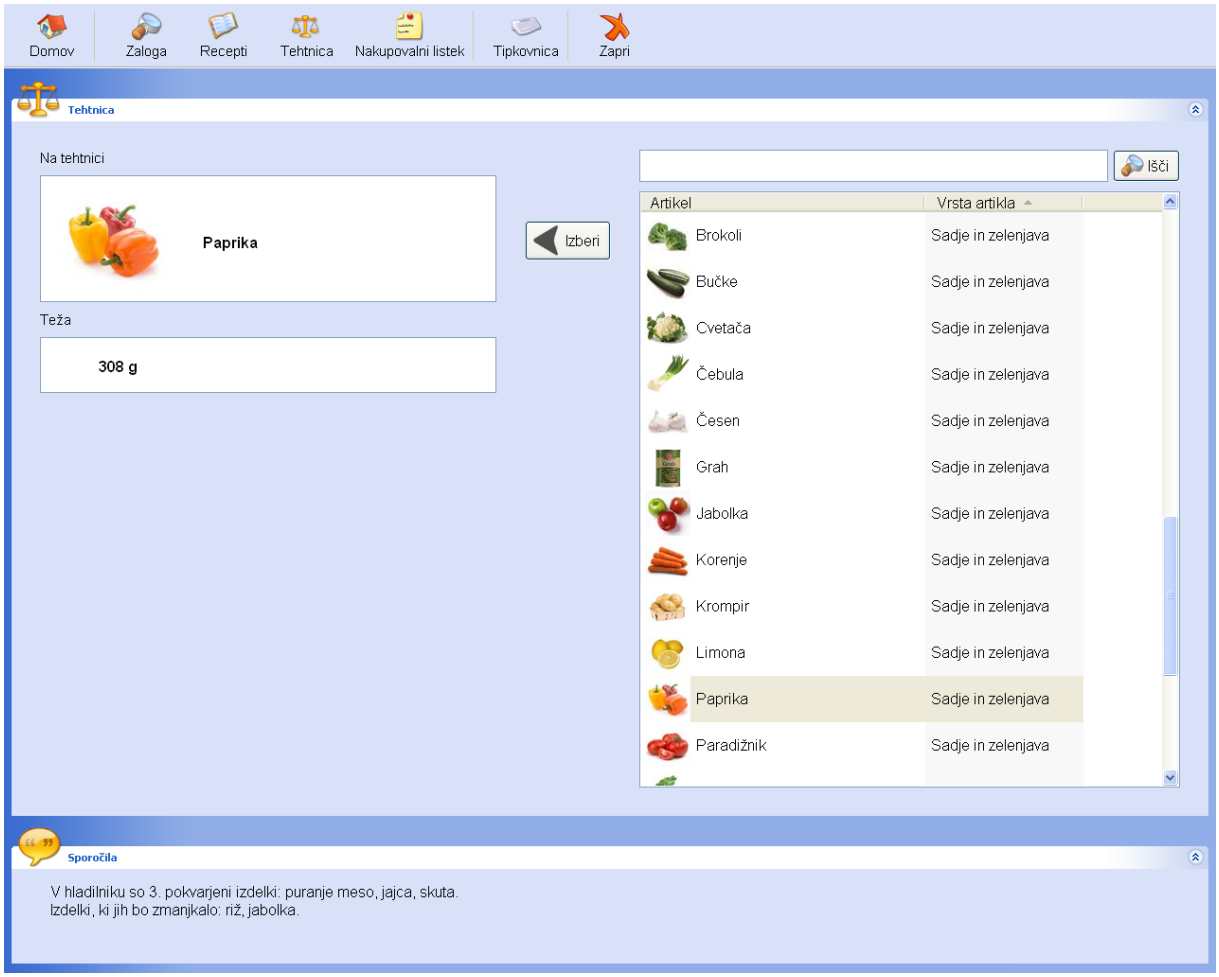
Slika 16: Pojavno okno za izbiranje med vsemi recepti.

Okno za izbiranje med vsemi recepti prikazuje slika 16. Na vrhu okna je orodna vrstica s petimi gumbi. Prvi gumb „Potrdi“ potrdi izbiro in zapre pojavno okno. Z gumboma „Nazaj“ in „Naprej“ se pomikamo med recepti. Gumb „Tipkovnica“ nam prikaže zaslonsko tipkovnico, gumb „Prekliči“ pa prekliče izbiro in zapre pojavno okno. Seznam vseh receptov je prikazan v tabeli. Recepte lahko sortiramo naraščajoče ali padajoče po imenu in številu artiklov, ki nam jih manjka za pripravo. Lahko jih tudi iščemo s pritiskom na gumb „Išči“. Recepti se iščejo po ključni besedi, zapisani v polju nad tabelo.

Posamezen recept pa se ob izbiri v tabeli prikaže s sliko in opisom točno tako, kot je že opisano v prejšnjem odstavku.

4.5.4 Zaslonska maska tehtnice

Slika 17 prikazuje zaslonsko masko za prikaz teže artikla na tehtnici.

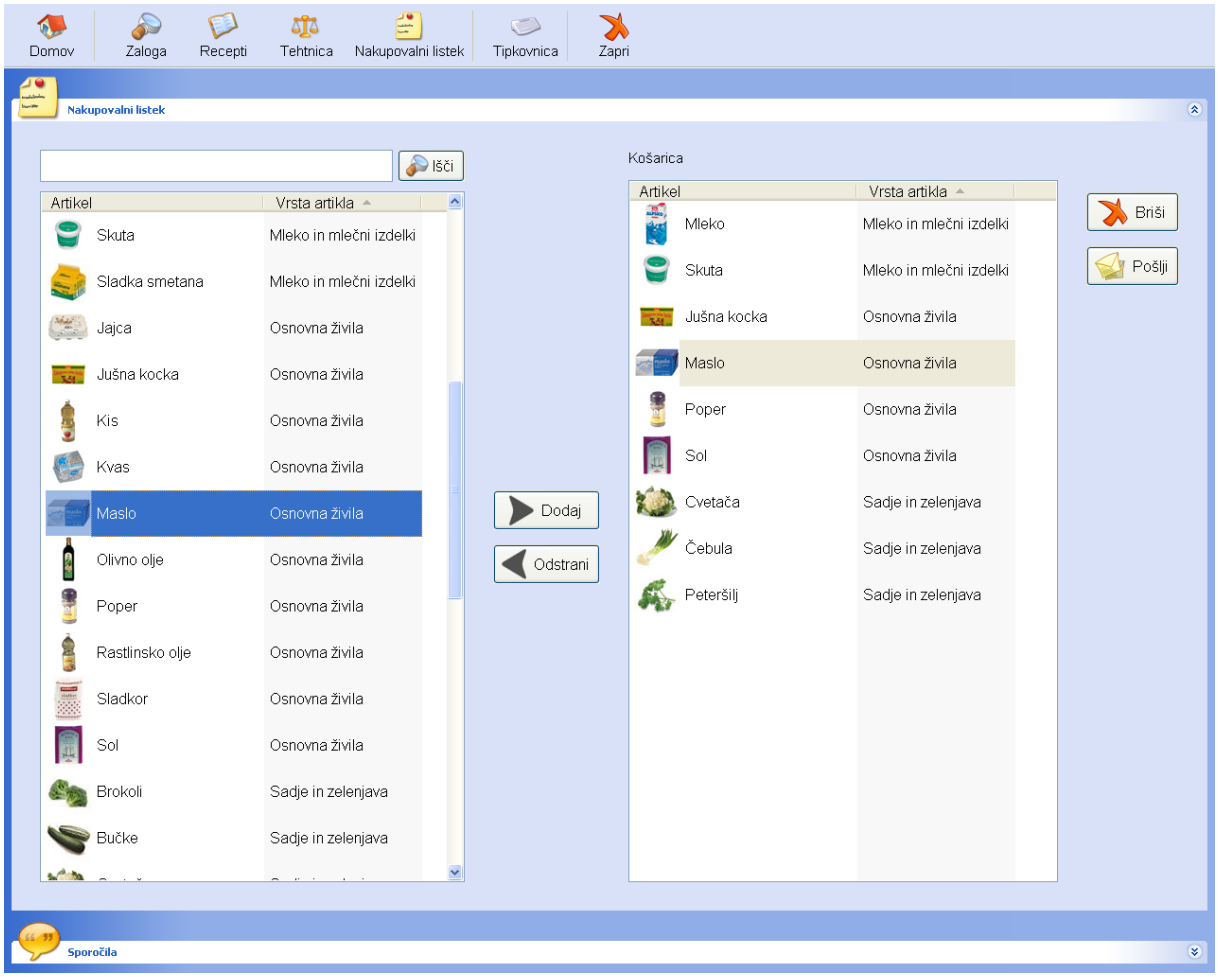


Slika 17: Zaslonska maska tehtnice.

Artikel, ki je postavljen na tehtnico, je prikazan s sliko in imenom v okvirju zgoraj levo. Pod njim pa je prikaza tehtana teža. Gumb „Izberi“ nam služi v primeru, da značka RFID na samem artiklu ne nosi podatka o artiklu ali pa se izkaže, da je ta podatek napačen. S pritiskom na ta gumb sprožimo zapis podatka o artiklu, ki smo ga izbrali v tabeli skrajno desno v značko RFID artikla, postavljenega na tehtnico. V desni tabeli je seznam vseh podprtih artiklov. Mogoče jih je sortirati po imenu in vrsti. Artikle pa lahko tudi iščemo po ključni besedi. Iskanje se izvede ob pritisku na gumb „Išči“.

4.5.5 Zaslonska maska nakupovalnega listka

Na naslednji sliki 18 je predstavljena zaslonska maska nakupovalnega listka.

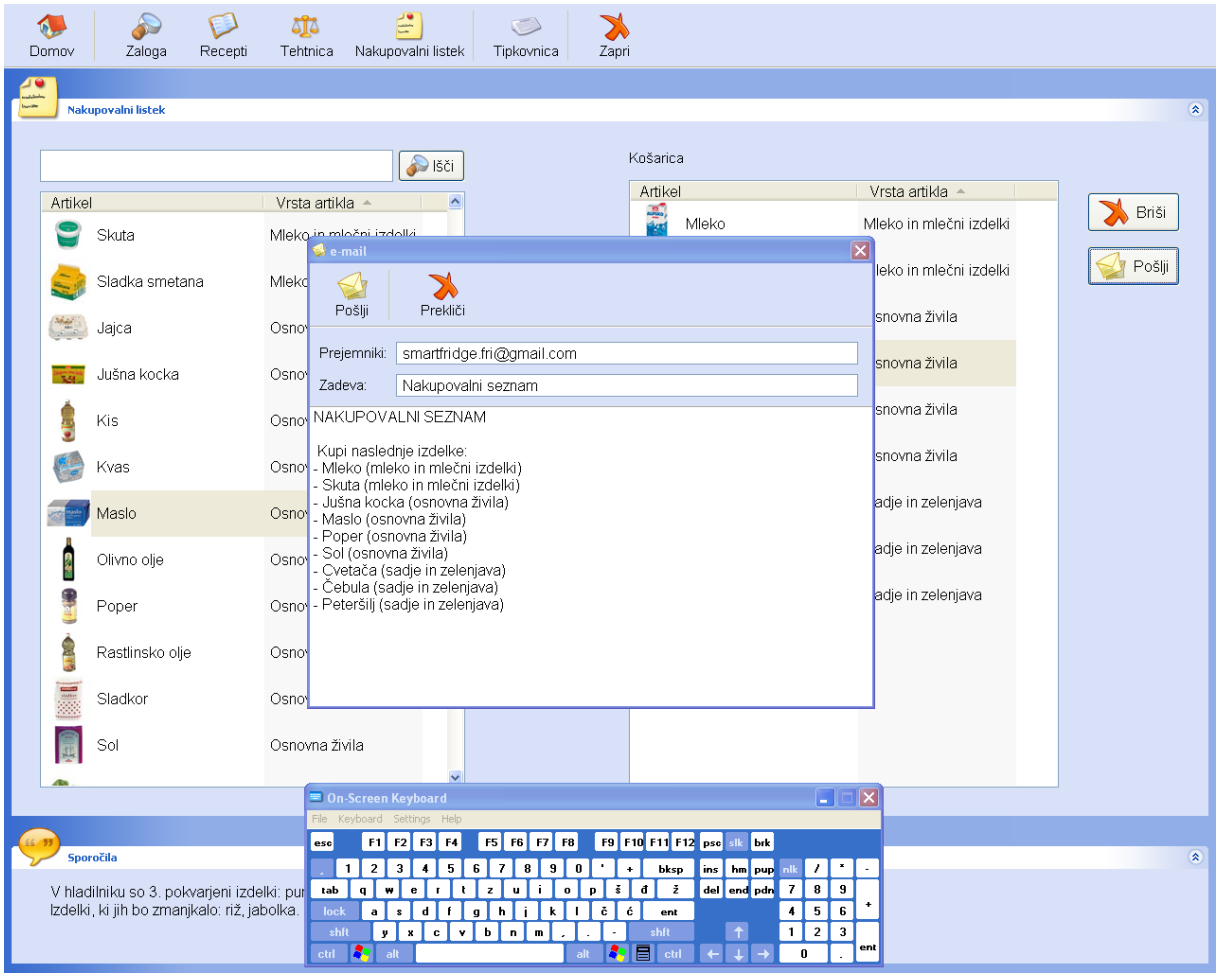


Slika 18: Zaslonska maska nakupovalnega listka.

V levi tabeli je prikazan seznam vseh artiklov, prav tako kot pri zaslonski maski tehtnice. Na sredini sta dva gumba, „Dodaj“ in „Odstrani“. Prvi doda izbran artikel iz leve tabele v desno oziroma v košarico, drugi pa izbrani artikel iz košarice odstrani. Na skrajni desni strani zgoraj sta še dva gumba. Prvi gumb, „Izbriši“, izbriše vse artikle iz košarice, drugi, „Pošlji“, pa prikaže novo okno za pošiljanje vsebine košarice kot nakupovalni seznam preko e-pošte.

Okno za pošiljanje e-pošte prikazuje slika 19. Na vrhu okna je orodna vrstica z dvema gumboma. Prvi gumb, „Pošlji“, pošlje elektronsko pošto, drugi, „Prekliči“, pa prekliče in zapre pojavno okno. Sledijo vnosna polja. V prvo polje vpišemo prejemnike, ločene z

vejicami. Drugo polje je namenjeno zadevi sporočila, zadnje pa sami vsebini sporočila. Tekstovni polji z zadevo in sporočilom sta predizpolnjeni, lahko pa jih seveda popravimo z uporabo zaslonske tipkovnice, ki je prikazana na dnu zaslona.



Slika 19: Pojavno okno za pošiljanje nakupovalnega listka.

4.5.6 Komunikacija s servisom

Aplikacija je s servisom povezana preko vtičnice na vratih 1899. Grafični vmesnik periodično posreduje zahtevke servisu. S temi zahtevki preverja, ali je prišlo do spremembe vsebine hladilnika in ali je na digitalno tehtnico postavljen kakšen artikel. V kolikor ugotovi spremembo zaloge, posodobi zaslonski maski z zalogo in predlogi obrokov. Ob detekciji artikla na tehtnici pa le-tega prikaže z njegovo pravkar stehtano težo na zaslonski maski tehtnice. Poleg teh periodičnih zahtevkov pa aplikacija pošilja servisu preko iste vtičnice tudi ukaz za prepis podatkov o artiklu na njegovi znački RFID. Ta ukaz se uporabi, le če uporabnik določi, da se prikazan artikel na zaslonski maski

tehtnice ne ujema z dejanskim.

4.6 Spletna aplikacija

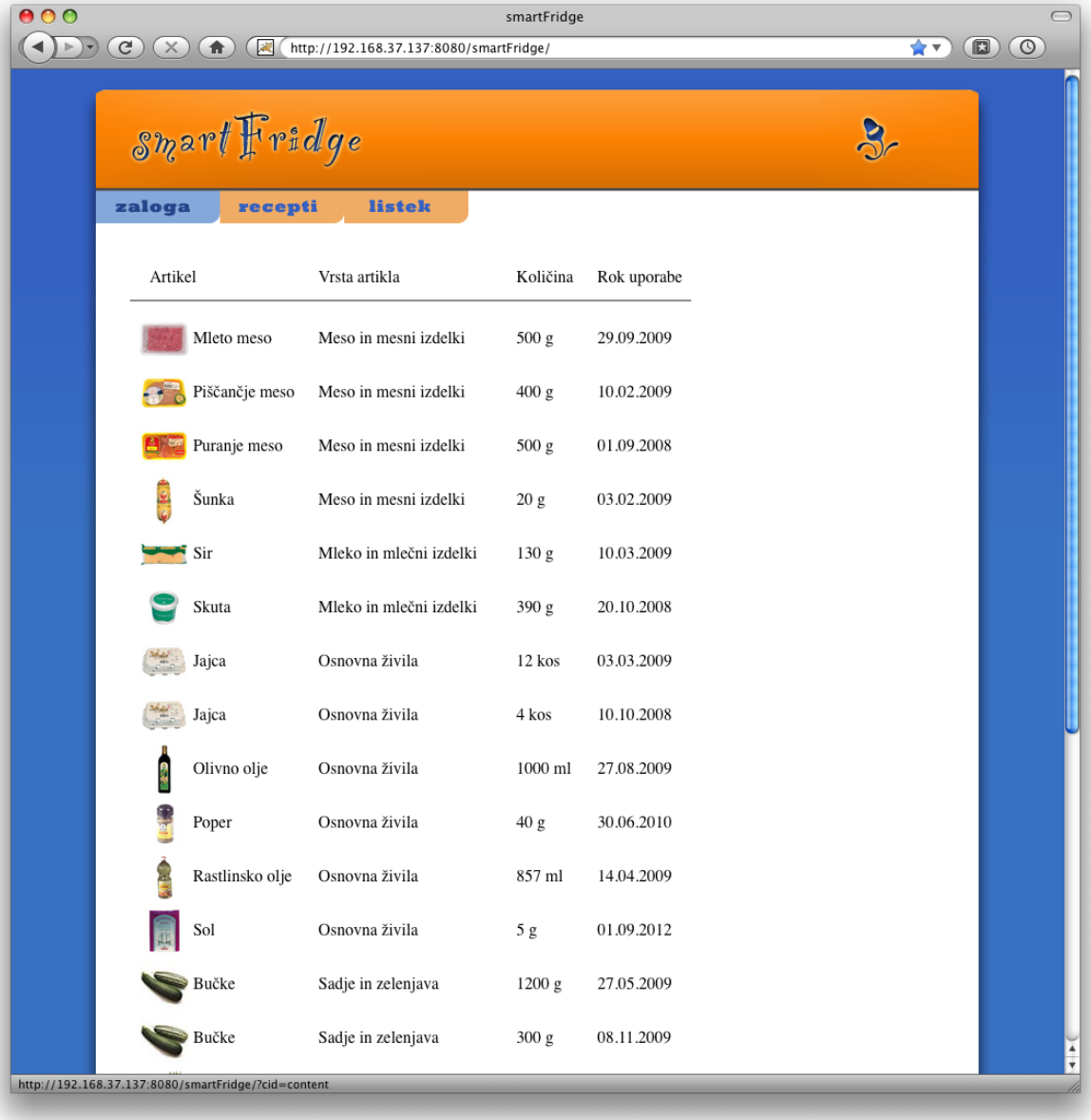
Spletna aplikacija je zgrajena na podlagi Javanske tehnologije JSP in omogoča dinamično generiranje spletnih strani. Spletna stran je zgenerirana v obliki dokumenta XHTML. Znotraj dokumenta pa sta uporabljena še mehanizma CSS in JavaScript, ki skrbita za dodatno pozicioniranje, izgled in interaktivnost posameznih elementov dokumenta. Pri gradnji dokumenta XHTML je bilo eksplicitno poskrbljeno, da so vsi elementi in njihovi atributi zapisani tako, da se pravilno prikažejo v večini današnjih spletnih brskalnikov. Preverjeni brskalniki so: Microsoft Internet Explorer, Mozilla Firefox, Safari, Opera ter Google Chrome [15].

Tehnologija JSP sloni na odprti, prostodostopni specifikaciji, ki jo je razvila družba Sun Microsystems. Prav zaradi dostopnosti specifikacije so se razvili še drugi odprtokodni produkti, povezani s to tehnologijo. Primer takega produkta je spletni aplikacijski strežnik Apache Tomcat, ki smo ga tudi uporabili za serviranje spletne aplikacije.

Osnovni namen spletne aplikacije je omogočiti dostop do nekaterih funkcionalnosti inteligentnega hladilnika iz oddaljenih lokacij. Te funkcionalnosti so: pregled vsebine hladilnika, pregled predlogov receptov ter pregled vsebine nakupovalnega listka.

4.6.1 Pregled zaloge

Slika 20 prikazuje pregled vsebine hladilnika preko spletnega odjemalca Mozilla Firefox na operacijskem sistemu Mac OSX. Izpis zaloge je privzeta stran spletne aplikacije ki se prikaže ob zagonu. Ponoven osvežen izpis sprožimo s pritiskom na gumb „zaloga“. Vsebina hladilnika je predstavljena kot seznam artiklov v tabeli.



Slika 20: Pregled vsebine hladilnika preko spletne aplikacije.

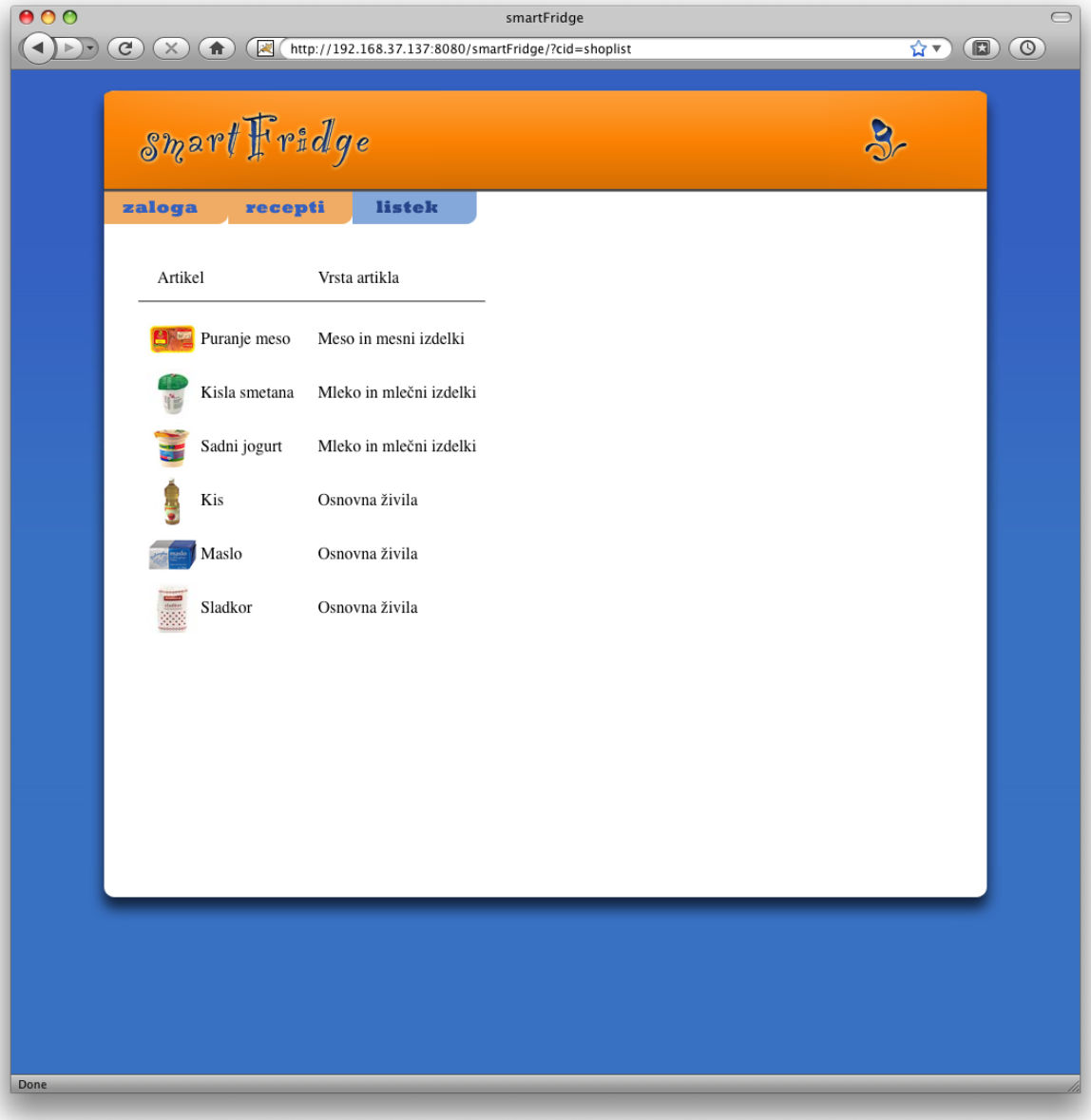
4.6.2 Pregled predlogov receptov

Pregled predlogov receptov posameznih obrokov predstavlja slika 21. Prikaz sprožimo s pritiskom na gumb „recepti“. Predlogi so predstavljeni po posameznih skupinah obrokov. Najprej so zgoraj predstavljeni predlogi predjedi, nato sledijo juhe in glavne jedi. Med predlogi posameznega tipa obroka se pomikamo s sledenjem povezavi „naprej“ oziroma „nazaj“.

Slika 21: Pregled predlogov receptov.

4.6.3 Pregled nakupovalnega listka

Pregled nakupovalnega listka prikazuje slika 22. Prikaz le-tega sprožimo s pritiskom na gumb „listek“. V tabeli so prikazani artikli, ki nam manjkajo in smo jih na grafičnem vmesniku hladilnika dodali v košarico za nakup.



Slika 22: Pregled nakupovalnega listka.

4.7 Namestitev

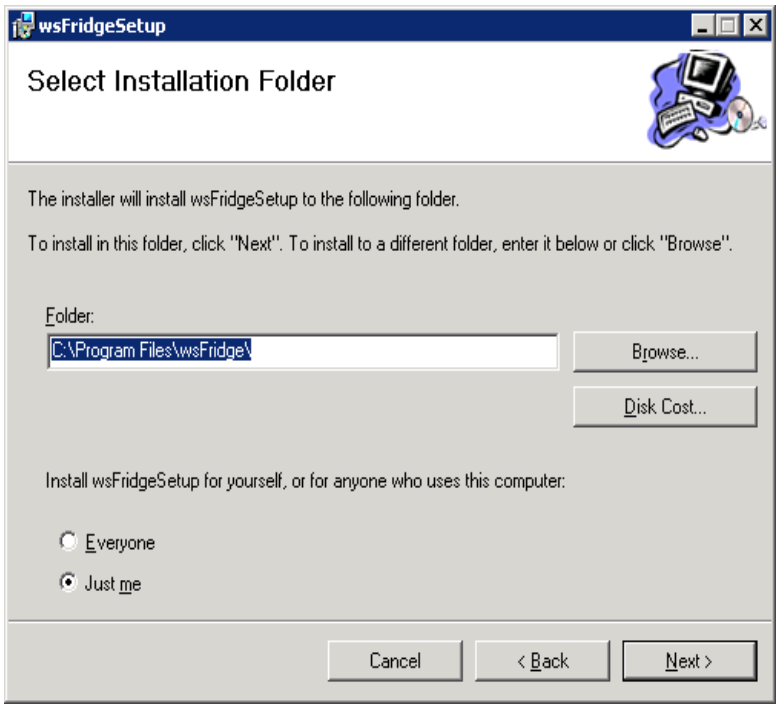
Predpogoj za delovanje opisanega sistema inteligentnega hladilnika je nameščen operacijski sistem Microsoft Windows verzije 2000 ali novejši. Poleg osnovne namestitve operacijskega sistema je treba namestiti še sledeče komponente:

- Microsoft .NET Framework verzije 3.5 ali novejši.
- Java Standard Edition Runtime Environment verzije 6 ali novejše.
- Podatkovno bazo PostgreSQL verzije 8.0 ali novejšo.
- Spletni aplikacijski strežnik Apache Tomcat verzije 6.0 ali novejši.

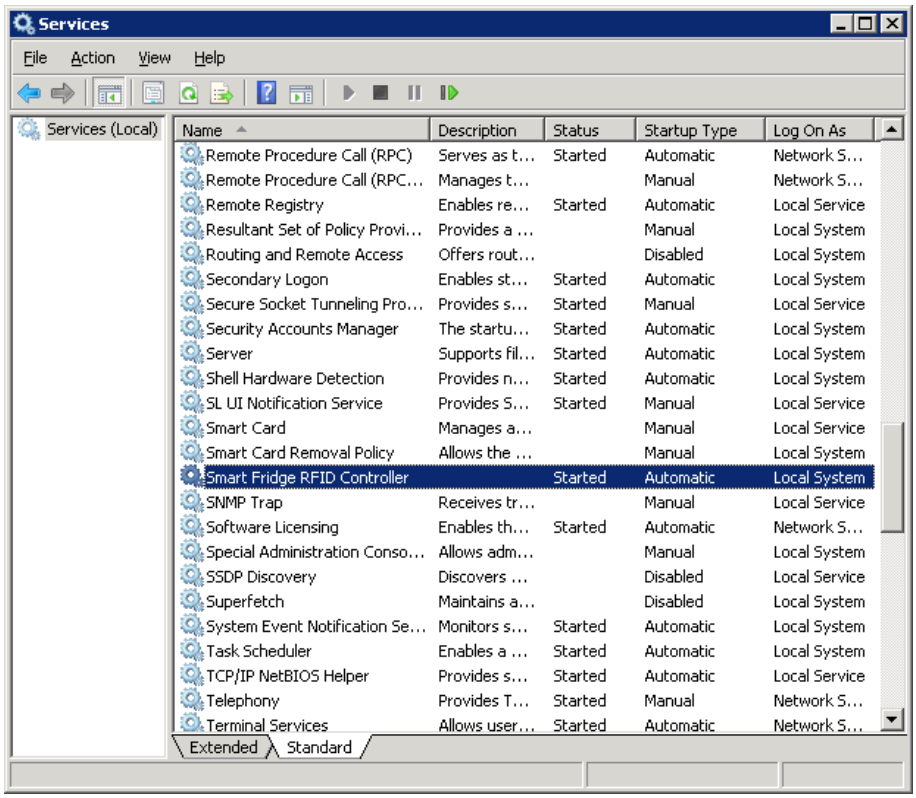
4.7.1 Namestitev servisa

Namestitveni program servisa sestoji iz dveh datotek: `wsFridgeSetup.msi` ter `setup.exe`. Postopek začnemo z zagonom izvršilne datoteke `setup.exe`. Namestitveni program nas v drugem koraku vpraša, v katero mapo naj razpakira pripadajoče datoteke. Slika 23 prikazuje primer izbire mape (`C:\Program Files\wsFridge\`). V vseh ostalih korakih imamo samo možnost napredovanja postopka, kar seveda potrdimo. Ob uspešni namestitvi je naša servisna aplikacija z imenom „Smart Fridge RFID Controller“ že registrirana med sistemskimi servisi. Za servis je privzeto, da se avtomatsko zažene ob zagonu operacijskega sistema. Lahko pa ga tudi ročno ustavimo ali ponovno zaženemo. To storimo z orodjem `services.msc`, ki je del operacijskega sistema in ga prikazuje slika 24.

Servis nima dostopa do uporabniškega profila, zato vse dogodke vpisuje v dnevniški datoteki `wsFridge.exe.log` in `wsFridge.exe.OLD.log`, ki se nahajata znotraj mape, ki smo jo izbrali pri namestitvi. Datoteki se rotirata, prva vsebuje najnovejše vnose, druga pa zagotavlja določeno mero zgodovine, ki je omejena na 1024 kB podatkov. Iz teh dveh datotek je mogoče diagnosticirati vzroke za morebitno napačno delovanje. Ti vzroki so tipično napačno vneseni parametri za dostop do podatkovne baze ali pa neodzivanje strojne opreme itn.



Slika 23: Namestitveni program Windows Servisa.



Slika 24: Upravljanje sistemskih servisov.

V mapi, ki smo jo izbrali v drugem koraku namestitve, se nahaja tudi datoteka `wsFridge.exe.config`. To je datoteka, namenjena konfiguraciji. Njena vsebina je:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <appSettings>

    <add key="RFIDReader Port" value="COM4" />
    <add key="RFIDReader BaudRate" value="115000" />
    <add key="RFIDReader FullPower" value="False" />

    <add key="Libela Port" value="COM1" />
    <add key="Libela BaudRate" value="9600" />
    <add key="LibelaRFIDReader Port" value="COM8" />
    <add key="LibelaRFIDReader BaudRate" value="115000" />
    <add key="LibelaRFIDReader FullPower" value="False" />

    <add key="PostgreSQL Server" value="localhost" />
    <add key="PostgreSQL Database" value="smart_fridge_db" />
    <add key="PostgreSQL User" value="postgres" />
    <add key="PostgreSQL Password" value="rfid" />

  </appSettings>
</configuration>
```

Nastaviti je treba naslednje parametre:

- RFIDReader Port: Ime RS-232 vrat, na katera je priključen čitalec RFID za detekcijo vsebine hladilnika.
- LibelaRFIDReader Port: Ime RS-232 vrat, na katera je priključen čitalec RFID za detekcijo artikla na tehtnici.
- Libela Port: Ime RS-232 vrat, na katera je priključena digitalna tehtnica.
- PostgreSQL Server: Naslov računalnika, na katerem teče podatkovna baza.
- PostgreSQL Database: Ime baze podatkov inteligentnega hladilnika.
- PostgreSQL User: Uporabniško ime za dostop do podatkovne baze.
- PostgreSQL Password: Geslo za dostop do podatkovne baze.

4.7.2 Namestitev aplikacije z grafičnim vmesnikom

Aplikacijo sestavljata dve datoteki: `smartFridge.conf` in `smartFridge.jar`. Ti dve datoteki samo skopiramo v poljubno mapo in s tem je namestitev zaključena.

Prva datoteka je namenjena konfiguraciji. Njena vsebina je:

```
#

dbConnectionString=jdbc:postgresql://localhost:5432/smart_fridge_db
dbUser=postgres
dbPass=rfid

picBaseDir=C:/slike/

fullScreen=true

webService=false
webServiceUrl=http://localhost:8080/axis2/services/wsPoisciRecepte
```

Pomen posameznih nastavitvenih parametrov pa je sledeč:

- dbConnectionString: Naslov URL za dostop do baze podatkov.
- dbUser: Uporabniško ime za dostop do podatkovne baze.
- dbPass: Geslo za dostop do podatkovne baze.
- picBaseDir: Mapa s slikami artiklov in receptov, na katere se sklicujejo podatki o artiklih in receptih v podatkovni bazi.
- fullScreen: S tem označimo, ali naj se aplikacija zažene v celozaslonskem načinu (true) ali pač ne (false).
- webService: S tem parametrom nastavimo, ali naj se recepti iščejo preko spletnega servisa (true) ali pa lokalno v podatkovni bazi (false).
- webServiceUrl: Naslov URL za dostop do spletnega servisa receptov.

Druga datoteka `smartFridge.jar` pa je Javanska izvršilna datoteka. Ob običajni inštalaciji okolja JRE to datoteko zaženemo z dvoklikom. V primeru, da akcija za dvoklik ni tako nastavljena, pa se program požene z ukazom `javaw -jar smartFridge.jar`.

4.7.3 Namestitev spletne aplikacije

Spletna aplikacija je zapakirana v datoteki `smartFridge.war`. Namestimo jo tako, da skopiramo to datoteko v mapo `CATALINA_HOME/webapps`, kjer `CATALINA_HOME` pomeni korensko mapo nameščenega spletnega strežnik Apache Tomcat. Poleg tega je treba na spletnem strežniku nastaviti parametre za dostop do podatkovne baze. To storimo z vpisom ustreznih vrednosti v datoteko `CATALINA_HOME/conf/context.xml`:

```
<Context>

    ...

    <Resource name="jdbc/postgresqlDB" auth="Container"
        type="javax.sql.DataSource"
        driverClassName="org.postgresql.Driver"
        url="jdbc:postgresql://localhost:5432/smart_fridge_db"
        username="postgres"
        password="rfid"
        maxActive="10"
        maxIdle="10"
        maxWait="-1" />

    ...

</Context>
```

Nastaviti je treba parametre:

- url: Naslov URL za dostop do baze podatkov.
- username: Uporabniško ime za dostop do podatkovne baze.
- password: Geslo za dostop do podatkovne baze.

Tako nameščena spletna aplikacija je dostopna na naslovu http://<strežnik>/smartFridge/.

5 Primerjava s klasičnim hladilnikom

Prototip inteligentnega hladilnika s priključenimi napravami, kot ga prikazuje slika 25, bi imel v primerjavi s klasičnim vrsto prednosti. V prvi skupini so to funkcije hladilnika, ki nas razbremenjujejo.

- Pregledi vsebine: Vsebina hladilnika je v našem primeru, prikazana v urejeni tabeli. Artikle lahko sortiramo po imenu, vrsti in datumu uporabnosti. Artikle lahko po slednjih atributih tudi iščemo in filtriramo. Tako v primeru, da želimo iz hladilnika odstraniti artikle, ki jim je pretekel rok uporabnosti, pritisnemo na gumb „Pokvarjeni“ na zaslonski maski vsebine. To nam izpiše vse pokvarjene artikle, ki jih lahko tudi fizično odstranimo iz hladilnika. V primeru klasičnega hladilnika bi za dosego istega cilja morali iz hladilnika vzeti posamezno vse artikle, najti na njih zapisan podatek o veljavnosti in na podlagi tega artikel zavreči ali pa ga vrniti nazaj v hladilnik. Slednji postopek je bistveno zamudnejši.

Slika 25: Realiziran prototip inteligentnega hladilnika.

- Obveščanje: Hladilnik nas obvešča o pokvarjenih in kmalu pokvarjenih izdelkih ter izdelkih z malo vsebine. Tako nam ni treba stalno preverjati vsebine, da nam na primer naslednji dan pri zajtrku ne bi zmanjkalo mleka.

- Predlogi: Hladilnik nam predlaga, katere jedi lahko sestavimo iz trenutne zaloge hladilnika.

- Oddaljeni pregledi: Vsebino hladilnika, predloge jedilnika in nakupovalni seznam lahko pregledujemo od daleč preko interneta. Tako lahko na primer pred odhodom iz službe pregledamo, katere stvari nam manjkajo, in jih na poti domov tudi kupimo. V primeru, da si lastimo naprednejši mobilni telefon ali kakšno podobno mobilno napravo, lahko nakupovalni seznam pregledujemo preko interneta tudi med samim nakupovanjem v trgovini.

Druga prednost je varčevanje z energijo. Klasični hladilnik moramo vsakič, ko želimo pogledati njegovo vsebino, tudi odpreti. Iz lastnih navad vem, da je teh odpiranj pred vsakim obrokom veliko. To pa pomeni konstantni odtok mrzlega zraka in posledično veliko porabo energije za vzdrževanje nastavljene temperature. Z uporabo inteligentnega hladilnika lahko ta odpiranja hladilnika zreduciramo na minimum.

Predstavljenih prednosti pa inteligentni hladilnik še ne bi mogel izkoristiti. Čeprav naj bi v prihodnjih letih na artiklih po policah trgovin nalepke RFID nadomestile črtne kode, tega danes še ni. Zato predstavljeni model hladilnika v realnem svetu še ni uporaben.

6 Sklepne ugotovitve

V sklopu diplomske naloge je bil razvit sistem inteligentnega hladilnika, ki omogoča avtomatsko vodenje vsebine hladilnika, prikaz vsebine, obvestil in predlogov. Izdelan je bil grafični vmesnik, prirejen za delo z zaslonom, občutljivim na dotik. Za oddaljen dostop do funkcionalnosti inteligentnega hladilnika pa je bila razvita še spletna aplikacija. Kljub temu je to le osnovni nabor funkcionalnosti, ki ga koncept inteligentnega hladilnika ponuja, zato je v nadaljevanju predstavljenih še nekaj idej za nadaljnje delo, ki lahko poteka na več področjih:

- Strojne izboljšave

  Za detekcijo artiklov v hladilniku bi lahko uporabili več čitalcev oziroma čitalec z več antenami. Tako bi izboljšali zanesljivost delovanja, saj je doseg čitalcev oziroma njihovih anten v najboljšem primeru le okoli pol metra. Poleg tega prihaja pri odzivu večjega števila značk na signal enega čitalca do kolizij, ki bistveno zmanjšajo zmogljivost sistema.

  Čitalca RFID v opisanem sistemu veš čas periodično preverjata, ali se je spremenilo stanje zaloge v hladilniku. Dejanska vsebina hladilnika pa se lahko spremeni le kadar hladilnik odpremo. Zato bi bilo smiselno v vrata vgraditi stikalo, s katerim bi vklapljali in izklapljali čitalec. Tako bi lahko stanje hladilnika preverjali le kratek čas po zaprtju vrat.

  Vsaka polica v hladilniku bi lahko bila tudi tehtnica, tako nam artiklov ne bi bilo treba postavljati na posebno območje za tehtanje.

- Podpora več platformam

  Kljub temu da smo pri razvoju težili k popolni platformski neodvisnosti sistema, nam to v celoti ni uspelo. Servis je namreč realiziran v Microsoftovi tehnologiji .NET. Poganjanje takih programov na ostalih platformah sicer omogoča projekt Mono, vendar pa ima ta produkt ravno v delu s komunikacijo z vodilom RS-232 še vedno neodpravljenega hrošča. Vseeno pa zaradi razlogov, navedenih že v poglavju o izbiri tehnologije, priporočam nadaljnji razvoj platformsko specifičnih servisov

še za ostale sisteme.

- Uporabniška izkušnja

  Spletna aplikacija ne ponuja vseh funkcij aplikacije z grafičnim vmesnikom na samem hladilniku. Manjkajo na primer iskanje, sortiranje in filtriranje artiklov v zalogi. Poleg tega bi bilo dobro imeti še preprostejšo verzijo za prikaz na mobilnih telefonih.

  Naslednja stvar je vpeljava umetne inteligence, tako da bi se hladilnik glede na preteklo spreminjanje vsebine in izbiro receptov naučil naših prehranjevalnih navad in bi v skladu s tem tudi deloval. Preprost primer tega bi bil, da bi hladilnik sam zaznal, da nam manjkajo sestavine za pripravo zajtrka, kakršnega jemo vsak dan, nas o tem opozoril in dodal manjkajoče sestavine v nakupovalno listo.

  Poleg tega je tukaj mogoče vpeljati še število novih funkcionalnosti, kot so: govorni ukazi, branje receptov oziroma glasovno vodenje kuhanja, video recepti, vodenje diete in podobo.

- Povezljivost z zunanjim svetom

  V sedanjem sistemu so artikli fiksno vneseni v podatkovno bazo. To bazo bi bilo smiselno povezati s posameznimi proizvajalci. Druga alternativa bi bila izdelava spletnega portala, kjer bi se vodila evidenca vseh artiklov in do katerega bi sistem hladilnika lahko dostopal. Tak enoten portal bi bil dobrodošel tudi za vodenje receptov.

7 Uporabljena literatura

[1] V. D. Hunt, A. Puglia, M. Puglia, RFID – A Guide to Radio Frequency Identification, New Jersy, Wiley-Interscience, 2007

[2] IDS d.o.o., IDS-R13MP Demonstration Board Data Sheet, 2006

[3] International Organization for Standardization, ISO 15693 – Identification cards – Contactless integrated circuit cards – Vicinity cards, 1999

[4] Philips Semiconductors, I-Code Sli Data Sheet – SL2 ICS20 Functional Specification, 2003

[5] Texas Instruments, TRF7960 User's Guide, 2007

[6] F. Thornton, B. Haines, A. M. Das, H. Bhargava, A. Campbell, J. Kleinschmidt, RFID Security, Syngress Publishing Inc., Rockland, 2006

[7] M. Umberger, I. Humar, „Pametne stavbe: Pametne stavbe so vse bolj priljubljene“, Finance.si, št. 20, 2006

[8] (2007) Innovation Lab and the Hacked, Intelligent Fridge. Dostopno na: http://www.nanovidensbank.dk/sw28520.asp

[9] (2007) Samsung Developing RFID Fridge. Dostopno na: http://www.guardian.co.uk/technology/blog/2007/jan/05/samsungdevelop

[10] (2008) Apache Tomcat. Dostopno na: http://tomcat.apache.org/index.html

[11] (2008) Electrolux - History time line. Dostopno na: http://www.electrolux.com/node304.aspx

[12] (2008) Evolution of the Fridge Computer. Dostopno na: http://www.orangecone.com/archives/2008/01/the_fridge_comp.html

[13] (2008) Introduction to Windows Service Applications. Dostopno na: http://msdn.microsoft.com/en-us/library/d56de412(VS.80).aspx

[14] (2008) Java EE Technologies at a Glance. Dostopno na: http://java.sun.com/javaee/technologies/index.jsp

[15] (2008) JavaServer Pages Technology. Dostopno na: http://java.sun.com/products/jsp/

[16] (2008) M. Rothensee, User Acceptance of the Intelligent Fridge: Empirical Results from a Simulation. Dostopno na: http://www.springerlink.com/content/978-3-540-78730-3/

[17] (2008) Radio-frequency identification – Wikipedia, the free encyclopedia. Dostopno na: http://en.wikipedia.org/wiki/RFID

[18] (2008) Refrigerator - Wikipedia, the free encyclopedia. Dostopno na: http://en.wikipedia.org/wiki/Refrigerator

[19] (2008) RFID – Portal Skupina RFID – vse o tehnologiji radijske identifikacije. Dostopno na: http://www.skupinarfid.com/index.php?stran=clanki

[20] (2008) Start – Mono. Dostopno na: http://mono-project.com/Start

[21] (2008) SWT: The Standard Widget Toolkit. Dostopno na: http://www.eclipse.org/swt/

[22] (2008) .NET Framework: Overview, dostopno na: http://www.microsoft.com/net/overview.aspx